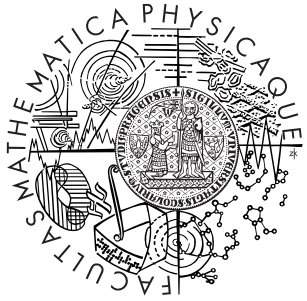
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE



Bc. Nataša Košová

Stochastické modely tvorby škodních rezerv

Katedra pravděpodobnosti a matematické statistiky

Vedoucí diplomové práce: RNDr. Ing. Iva Justová, Ph.D.

Studijní program: Matematika
Studijní obor: Finanční a pojistná matematika

Praha 2011

Na tomto mieste by som rada poďakovala hlavne vedúcej mojej diplomovej práce RNDr. Ing. Ive Justovej, Ph.D. za vedenie a konzultácie pri písaní tejto práce. Tiež by som chcela poďakovať svojim rodičom za morálnu a finančnú podporu počas celého trvania štúdia na MFF UK.

Prehlasujem, že som svoju diplomovú prácu vypracovala samostatne a výhradne s použitím citovaných prameňov, literatúry a ďalších odborných zdrojov.

Beriem na vedomie, že sa na moju prácu vzťahujú práva a povinnosti vyplývajúce zo zákona č. 121/2000 Sb., autorského zákona v platnom znení, najmä skutočnosť, že Univerzita Karlova v Prahe má právo na uzavretie licenčnej zmluvy o použití tejto práce ako školského diela podľa §60 odst. 1 autorského zákona.

V Prahe dňa 6. decembra 2011 Nataša Košová

Názov práce: Stochastické modely tvorby škodních rezerv
Autor: Bc. Nataša Košová
Katedra (ústav): Katedra pravděpodobnosti a matematické statistiky
Vedúci diplomovej práce: RNDr. Ing. Iva Justová, Ph.D.
e-mail vedúceho: iva_justova@hotmail.com

Abstrakt: V predloženej práci študujeme a popisujeme stochastický model vytvárania škodných rezerv pre poisťovne. Konkrétne sa jedná o model založený na nasledujúcich troch vlastnostiach. Modelovanie očakávaných poistných plnení je závislé na neznámych parametroch, ktoré je nutné čo najpresnejšie odhadnúť. Úhrny nastaných a vyplatených škôd za konkrétne roky ich vzniku a následného plnenia sa riadia kolektívnym modelom rizika. Konečná výška rezervy je odhadnutá bayesovskou metodológiou, ktorá používa apriórne informácie od väčšieho počtu poisťovní. Súčasťou práce je aj vytvorenie funkčného programu na výpočet škodných rezerv daným modelom a jeho nasledovné otestovanie na simulovaných dátach.

Kľúčové slová: škodná rezerva, metóda Cape Cod, kolektívny model rizika, Bayesova veta

Title: Stochastic Loss Reserving Models
Author: Bc. Nataša Košová
Department: Department of Probability and Mathematical Statistics
Supervisor: RNDr. Ing. Iva Justová, Ph.D.
Supervisor's e-mail address: iva_justova@hotmail.com

Abstract: In present thesis we study and describe a stochastic loss reserve model for individual insurers. Specifically, it is the model based on the three following features. Modelling of expected claims depends on unknown parameters which estimates need to be the most accurate. Aggregated occurred and paid losses for particular years are modelled by a collective risk model. The final reserve is estimated by Bayesian methodology that uses a prior information from a significant number of insurers. Part of the thesis is also an implementation of the program that calculates reserves by using our model and its testing on simulated data.

Keywords: loss reserve, Cape Cod method, collective risk model, Bayes theorem

Obsah

1 Úvod **1**

2 CNB model **3**

2.1 Analyzované dáta 3

2.2 Metóda Cape Cod 4

2.3 Kolektívny model rizika 6

2.4 Podmienené CNB rozdelenie 11

2.5 Odhad neznámych parametrov 16

2.6 Bayesova veta 19

2.7 Škodná rezerva 23

3 Simulácia modelu **25**

3.1 Generovanie vývojových trojuholníkov 25

3.2 Funkcia LAS 28

3.3 Optimalizácia a odhad parametrov 30

3.4 Bayesovský odhad škodnej rezervy 36

3.5 Porovnanie výsledkov CNB modelu a metódy Chain–Ladder 38

4 Záver **43**

Literatúra **45**

Prílohy **46**

Kapitola 1

Úvod

V neživotnom poistení často dochádza k tomu, že medzi vznikom poistnej udalosti a vyplatením poistného plnenia je významný časový posun. To je dôvod, prečo poisťovne majú povinnosť vytvárať tzv. *rezervy na poistné plnenia*, označované aj ako škodné rezervy.

Škodná rezerva sa v praxi typicky stanovuje vo výške súčtu dvoch hodnôt a to odhadov rezerv *RBNS* a *IBNR*. Názov rezervy RBNS je skratkou z anglického spojenia „reported but not settled“, čo napovedá, že výška tejto rezervy pokrýva poistné plnenia za škody poisťovni nahlásené, ale doposiaľ nezlikvidované. Rezerva IBNR je určená k pokrytiu poistných plnení za škody v minulosti vzniknuté, ale doposiaľ nenahlásené, čomu odpovedá jej názov „incurred but not reported“.

Rezerva RBNS je tvorená odhadmi, ktoré určujú likvidátori zvlášť pre jednotlivé poistné udalosti. Naproti tomu výška IBNR rezervy sa stanovuje pomocou matematicko-štatistických metód (napr. Chain–Ladder, Bornheutter–Fergusonova metóda, ...). Tieto metódy odhadujú výšku rezervy na základe informácií o minulých poistných plneniach, ktoré sú reprezentované buď kumulatívnymi, alebo nekumulatívnymi vývojovými trojuholníkmi.

V tejto práci sa zameriame práve na tvorbu zmieňovanej IBNR rezervy tzv. stochastickou metódou. Stochastická metóda alebo model je nástroj na odhadnutie rozdelenia pravdepodobnosti s pripustením jednej, alebo viacerých náhodných premenných. Náhodná premenná je obvykle založená na kolísaní pozorovaných hodnôt v historických dátach. Rozdelenie je potom odvodené z veľkého množstva simulácií, ktoré odrážajú túto náhodnosť. Konkrétne budeme v práci študovať stochastický

CNB model (z anglického Compound Negative Binomial distribution - zložené negatívne binomické rozdelenie). Jedná sa o model, ktorý vytvoril Glenn G. Meyers použitím veľkého množstva dát od viac ako 200 poisťovní a prvýkrát ho predstavil v článku [5]. Vďaka tomu, že mal k dispozícii také množstvo trénovacích dát, bol schopný vytvoriť model, ktorý využíva bayesovskú metodológiu, kedy sa model „natrénuje“ na dátach a tým dospeje k čo najpresnejšiemu výsledku.

V nasledujúcej kapitole popíšeme podrobne teóriu a postupy používané v CNB modely. Jedná sa o zložitú metódu, ktorá využíva netriviálne matematické postupy. Najprv upresníme, pre aké dáta je model vhodný a popíšeme základný model Cape Cod, ktorý bude popisovať očakávané úhrny škôd. V ďalšom popíšeme rozdelenie výšky a počtu škôd, keďže úhrny škôd sa budú riadiť kolektívnym modelom rizika. Uvedieme postup pre odhadnutie neznámych parametrov v Cape Cod modele, čomu bude predchádzať teoretický popis diskretizácie rozdelenia výšky škôd a rýchlej Fourierovej transformácie. Nakoniec vysvetlíme ako s použitím Bayesovej vety dospieť ku konečnej výške rezervy.

V tretej kapitole sa už budeme venovať praktickému použitiu skôr popísaného modelu. Súčasťou práce bude funkčný program simulujúci CNB model na tvorbu škodných rezerv. Hlavnou časťou kapitoly bude postup pri vytváraní programu a popis zdrojového kódu. Predtým ešte uvedieme na akých dátach budeme model testovať a za tým učelom vytvoríme generátor vývojových trojuholníkov. Nakoniec spočítame výšku rezervy známou metódou Chain–Ladder a tento výsledok porovnáme s výsledkom pri použití CNB modelu.

V poslednej kapitole, a teda v závere práce, zhrnieme výsledky, ku ktorým sme sa v jej priebehu dopracovali.

Kapitola 2

CNB model

Bayesovský CNB model je stochastický model, prostredníctvom ktorého dokážeme odhadnúť rozdelenie budúcich poistných plnení a následne aj výšku škodnej rezervy. K jeho použitiu je potrebné poznať viacero matematických postupov a teórií, ktoré postupne v práci popíšeme. Ešte predtým ale objasníme, pre aké dáta je model vhodný.

2.1 Analyzované dáta

Dáta potrebné k odhadu výšky rezervy musia obsahovať informácie o výškach nastaných škôd, prípadne vyplatených poistných plneniach. Tieto dáta by mali byť usporiadané do tzv. nekumulatívneho vývojového trojuholníka. Rozdiel medzi kumulatívnym a nekumulatívnym trojuholníkom je v tom, že v kumulatívnom trojuholníku sa výšky poistných plnení za jeden rok postupne sčítajú po vývojových rokoch, zatiaľ čo v nekumulatívnom trojuholníku uvádzame len prírastkové hodnoty.

Vývojový trojuholník uvedený v tabuľke 2.1. obsahuje informácie o úhrne škôd vzniknutých v roku i a uhradených v $j-$tom roku od vzniku škody, označených $X_{i,j}$, kde $i = 1, 2, \cdots, m$ a $j = 1, 2, \cdots, m - i + 1$. Budeme teda predpokladať, že vývoj škôd monitorujeme do m rokov od vzniku škody.
Rok vzniku (accident year) i označuje rok, kedy došlo k vzniku poistnej udalosti.
Vývojový rok (development/settlement year) j určuje počet rokov od vzniku poistnej udalosti, kedy bolo poistné plnenie za túto škodu uhradené.

Rok vzniku	**Poistné**	**Vývojový rok**						
		1	**2**	**...**	**j**	**...**	**m-1**	**m**
1	P_1	$X_{1,1}$	$X_{1,2}$	$\cdots$	$X_{1,j}$	$\cdots$	$X_{1,m-1}$	$X_{1,m}$
2	P_2	$X_{2,1}$	$X_{2,2}$	$\cdots$	$X_{2,j}$	$\cdots$	$X_{2,m-1}$	
$\vdots$	$\vdots$							
i	P_i	$X_{i,1}$	$X_{i,2}$	$\cdots$	$X_{i,j}$	$\cdots$		
$\vdots$	$\vdots$							
$m-1$	P_{m-1}	$X_{m-1,1}$	$X_{m-1,2}$					
m	P_m	$X_{m,1}$						

Tabuľka 2.1: Nekumulatívny vývojový trojuholník

Poistným (premium) sa myslí zaslúžené poistné za rok vzniku škody.

Poznámka (výpočet rezervy): Jednoducho povedané, výpočet škodnej rezervy znamená, že vývojový trojuholník doplníme na štvorec odhadmi budúcich úhrnov škôd. Výsledna rezerva vznikne sčítaním týchto hodnôt, teda

$$\text{rezerva} = \sum_{i=2}^{m} \sum_{j=m+2-i}^{m} \hat{X}_{i,j}.$$

2.2 Metóda Cape Cod

Prvý krok potrebný k určovaniu a odhadovaniu očakávaných škôd je výber základnej metódy pre ich modelovanie. Očakávané škody v nami popisovanom CNB modely určíme pomocou tzv. *Cape Cod metódy*. Ide o metódu nezávisle vyvinutú vedcami Jamesom Stanardom a Hansom Bühlmannom, a preto býva v literatúre uvedená aj pod názvom Stanard–Bühlmannova metóda. Na rozdiel od známejšej metódy Chain–Ladder berie do úvahy mimo vývojových koeficientov aj tzv. škodný pomer, ktorý vyjadruje vzťah medzi vzniknutou škodou a zaslúženým poistným. Očakávané škody modelované Cape Cod metódou majú tvar

$$E[X_{i,j}] = P_i \cdot L \cdot F_j, \tag{2.1}$$

kde P_i je poistné pre daný rok vzniku škody, L je označenie pre očakávaný škodný pomer a F_j je vývojový faktor závislý na vývojovom roku.

Parametre L a $\{F_j\}$ sú neznáme, a preto musia byť odhadnuté z poskytnutých dát. Na odhad oboch parametrov použijeme metódu maximálnej vierohodnosti. Táto metóda bude popísaná nižšie v samostatnej podkapitole, pretože jej aplikácia v prípade CNB modelu je zložitejšia. Pre zaujímavosť uvedieme v poznámke jednoduchý spôsob odhadu neznámych parametrov použiteľný pre kumulatívne trojuholníky.

Poznámka (odhad parametrov v metóde Cape Cod pre kumulatívne trojuholníky): Medzi škodami v kumulatívnom trojuholníku a prírastkovými škodami v nekumulatívnom trojuholníku existuje jednoduchý vzťah prevádzania jedného typu na druhý. Je preto v konečnom dôsledku jedno, ktorý typ vývojového trojuholníka modelujeme. Nech $Y_{i,j}$ sú kumulované škody a $X_{i,j}$ sú škody prírastkové. Potom platí

$$Y_{i,j} = \sum_{k=1}^{j} X_{i,k}$$

a naopak

$$X_{i,j} = \begin{cases} Y_{i,j} & \text{ak } j = 1, \\ Y_{i,j} - Y_{i,j-1} & \text{inak.} \end{cases}$$

Jeden z popisov metódy Cape Cod aplikovanej na kumulatívne trojuholníky je uvedený v [1], str. 201. Popísaný model predpokladá, že poznáme výšku poistného P_i pre jednotlivé roky vzniku poistných udalostí a ich kumulatívne hodnoty za jednotlivé obdobia $Y_{i,j}$. Škodný pomer L sa berie do úvahy dlhodobý (jednotný pre všetky roky trvania poistenia) a je určený podielom celkového poistného plnenia a celkového poistného za všetky pozorované roky. Poistné je ešte vynásobené o tzv. inverzné koeficienty, aby bolo porovnateľné s poistným plnením. Tieto koeficienty sú inverznými hodnotami k vývojovým koeficientom f_j metódy Chain–Ladder spočítané ako

$$\begin{aligned} f_{m-j} &= \frac{\sum_{i=1}^{m-j} Y_{i,j+1}}{\sum_{i=1}^{m-j} Y_{i,j}}, \quad j = 1, \cdots, m-1, \\ f_m &= 1. \end{aligned}$$

Odhad úhrnu škôd metódou Cape Cod pre kumulatívne trojuholníky má teda tvar

$$\hat{Y}_{i,j} = P_i \cdot \hat{L} \cdot \left(1 - \frac{1}{f_j}\right),$$

kde

$$\hat{L} = \frac{\sum_{i=1}^{m} Y_{i,m-i+1}}{\sum_{i=1}^{m} P_i f_i}$$

a odhady vývojových faktorov $\{F_j\}$ pre Cape Cod majú tvar

$$\hat{F}_j = 1 - \frac{1}{f_j}, \quad j = 1, \cdots, m.$$

2.3 Kolektívny model rizika

Základnou metódou modelovania rizika je kolektívny model. Ten nachádza uplatnenie v širšom popise činností poisťovní, základom ktorého je modelovanie celkového úhrnu škôd za dané obdobie. Podkladovou metódou CNB modelu je práve kolektívny model rizika, ktorý k dosiahnutiu rozdelenia celkového úhrnu škôd kombinuje rozdelenie výšky a počtu škôd.

Predpokladajme, že úhrn škôd pre špecifický rok vzniku a vývoja $X_{i,j}$ je náhodná veličina. Tá sa riadi kolektívnym modelom rizika, ktorý je popísaný nasledovne:

- Nech $N_{i,j}$ je náhodná veličina, ktorá reprezentuje počet škôd vzniknutých v roku i a uhradených v j–tom roku po ich vzniku.
- Nech Z_j je náhodná veličina vyjadrujúca výšku škody, ktorej rozdelenie je závislé na roku vývoja j a $\{Z_{j,k}\}_{k=1,2,\cdots}$ je postupnosť rovnako rozdelených náhodných veličín.
- Potom definujeme veličinu $X_{i,j}$ ako súčet $N_{i,j}$ škôd výšky Z_j. To znamená, že platí

  $$X_{i,j} = \sum_{k=1}^{N_{i,j}} Z_{j,k}$$

  a špeciálne, ak $N_{i,j} = 0$, potom aj $X_{i,j} = 0$.

V CNB modely majú počty škôd negatívne binomické (NB) rozdelenie, ktorého vyjadrenie a parametrizáciu dostaneme ako zmes gama a Poissonovho rozdelenia.

- Nech χ je nezáporná náhodná veličina s gama rozdelením takým, že $E[\chi] = 1$ a $Var[\chi] = c$, tj. $\chi \sim \text{Gama}(\frac{1}{c}, c)$.

- Nech $N_{i,j}$ je náhodná veličina počtu škôd a nech podmienené rozdelenie $N_{i,j}$ za podmienky χ má Poissonovo rozdelenie so strednou hodnotou $\chi\lambda_{i,j}$, tj. $N_{i,j}|\chi \sim \text{Poiss}(\chi\lambda_{i,j})$.

- Potom $N_{i,j}$ má negatívne binomické rozdelenie, pretože postupným výpočtom dostaneme

$$\begin{aligned} P(N_{i,j} = n) &= \int_0^\infty f_{N_{i,j},\chi}(n,x)dx = \int_0^\infty f_{N_{i,j}|\chi}(n) \cdot f_\chi(x)dx = \\ &= \int_0^\infty \frac{(x\lambda_{i,j})^n e^{-x\lambda_{i,j}}}{n!} \cdot x^{\frac{1}{c}-1} \frac{e^{-\frac{x}{c}}}{c^{\frac{1}{c}}\Gamma(\frac{1}{c})} dx = \\ &= \frac{\lambda_{i,j}^n}{\Gamma(n+1)\Gamma(\frac{1}{c})c^{\frac{1}{c}}} \underbrace{\int_0^\infty x^{n+\frac{1}{c}-1} e^{-x\left(\lambda_{i,j}+\frac{1}{c}\right)} dx}_{\text{ozn.} I} = (*). \end{aligned}$$

  Integrál I spočítame pomocou gama funkcie, ktorá je definovaná pre komplexné číslo a také, že $Re(a) > 0$ vzťahom $\Gamma(a) = \int_0^\infty t^{a-1}e^{-t}dt$. Keď označíme $a = n + \frac{1}{c}$ a $b = \lambda_{i,j} + \frac{1}{c}$, tak dostaneme

$$\begin{aligned} I &= \int_0^\infty x^{a-1}e^{-xb}dx = |\text{ subst. } t = xb\ | = \\ &= \int_0^\infty \left(\frac{t}{b}\right)^{a-1} e^{-t}\frac{dt}{b} = \\ &= \frac{1}{b^a}\int_0^\infty t^{a-1}e^{-t}dt = \\ &= \frac{\Gamma(a)}{b^a} \end{aligned}$$

  a po dosadení za premenné a, b máme $I = \frac{\Gamma\left(n+\frac{1}{c}\right)}{\left(\lambda_{i,j}+\frac{1}{c}\right)^{n+\frac{1}{c}}}$. Nakoniec, po dosadení

tohto výsledku do pôvodného výpočtu dostaneme

$$\begin{aligned}
(*) &= \frac{\lambda_{i,j}^n}{\Gamma(n+1)\Gamma\left(\frac{1}{c}\right)c^{\frac{1}{c}}} \cdot \frac{\Gamma\left(n+\frac{1}{c}\right)}{\left(\lambda_{i,j}+\frac{1}{c}\right)^{n+\frac{1}{c}}} = \\
&= \frac{\Gamma\left(n+\frac{1}{c}\right)}{\Gamma(n+1)\Gamma\left(\frac{1}{c}\right)} \cdot \frac{\lambda_{i,j}^n}{c^{\frac{1}{c}}\left(\lambda_{i,j}+\frac{1}{c}\right)^{n+\frac{1}{c}}} = \\
&= \frac{\left(n+\frac{1}{c}-1\right)!}{n!\left(\frac{1}{c}-1\right)!}\left(\frac{1}{c\left(\lambda_{i,j}+\frac{1}{c}\right)}\right)^{\frac{1}{c}}\left(\frac{\lambda_{i,j}}{\lambda_{i,j}+\frac{1}{c}}\right)^n = \\
&= \binom{n+\frac{1}{c}-1}{n}\left(\frac{1}{c\lambda_{i,j}+1}\right)^{\frac{1}{c}}\left(1-\frac{1}{c\lambda_{i,j}+1}\right)^n.
\end{aligned}$$

Takže vidíme, že počet škôd $N_{i,j}$ má naozaj negatívne binomické rozdelenie so strednou hodnotou

$$E\left[N_{i,j}\right] = E_\chi\left[E\left[N_{i,j}|\chi\right]\right] = E_\chi\left[\chi\lambda_{i,j}\right] = \lambda_{i,j}E_\chi\left[\chi\right] = \lambda_{i,j}$$

a rozptylom

$$\begin{aligned}
Var\left[N_{i,j}\right] &= E_\chi\left[Var\left[N_{i,j}|\chi\right]\right] + Var_\chi\left[E\left[N|\chi\right]\right] = \\
&= E_\chi\left[\chi\lambda_{i,j}\right] + Var_\chi\left[\chi\lambda_{i,j}\right] = \\
&= \lambda_{i,j} + \lambda_{i,j}^2 Var_\chi\left[\chi\right] = \lambda_{i,j} + \lambda_{i,j}^2 c.
\end{aligned}$$

Poznámka (parametrizácia NB rozdelenia): Obvykle sa stretávame s NB rozdelením v tvare

$$P(N=n) = \binom{r+n-1}{n} p^r (1-p)^n, n = 0, 1, 2, \cdots,$$

kde $r > 0$ a $0 < p < 1$. Keď dáme do rovnosti strednú hodnotu a rozptyl oboch zmienených parametrizácií, teda

$$\lambda_{i,j} = \frac{r(1-p)}{p}$$

a

$$\lambda_{i,j} + \lambda_{i,j}^2 c = \frac{r(1-p)}{p^2},$$

dostaneme $p = \frac{1}{1+c\lambda_{i,j}}$ a $r = \frac{1}{c}$. Pre takto definované parametre negatívne binomického rozdelenia dostaneme jeho vyššie odvodenú parametrizáciu, kde $\lambda_{i,j} > 0$ a $c > 0$.

Stredná hodnota $\lambda_{i,j}$ je očakávaným počtom škôd pre konkrétne roky i a j a pre už uhradené škody si zadefinujeme tento počet ako podiel očakávaných celkových úhrnov a očakávanej výšky škôd, tj.

$$\lambda_{i,j} = \frac{E[X_{i,j}]}{E[Z_j]} = \frac{P_i \cdot L \cdot F_j}{E[Z_j]}. \tag{2.2}$$

Parameter c sa v odborných textoch pomenúva ako tzv. *contagion parameter*. Týmto parametrom a jeho hodnotou sa bližšie zaoberá Meyers v [7] . Podľa tohto článku je možné parameter c interpretovať ako mieru neistoty a nekonzistentnosť počtu škôd.

Okrem rozdelenia počtu škôd, potrebujeme poznať aj rozdelenie výšky škod Z_j. Pri stanovení vhodného rozdelenia je potrebné zobrať do úvahy najmä dve veci:

1. Výšky jednotlivých škôd môžu nadobúdať len nezáporné hodnoty.
2. Výskyt malých škôd je zreteľne väčší ako tých veľkých.

Inými slovami, potrebujeme zvoliť také rozdelenie, ktoré je nezáporné a má značne veľkú šikmosť. Najčastejšie sa z týchto dôvodov používa exponenciálne, Paretovo, prípadne gama rozdelenie. Druhou možnosťou je vypočítať konkrétne hodnoty distribučnej funkcie výšky škôd z napozorovaných historických dát. Vzhľadom k tomu, že väčšinou ale nie je možné dostať sa k relevantnému množstvu dát, z ktorého by sme boli schopný určiť tabuľku hodnôt distribučnej funkcie, budeme pre výšky škôd predpokladať Paretovo rozdelenie.

Paretovo rozdelenie, na rozdiel od rozdelenia exponenciálneho, berie do úvahy možnosť výskytu značne väčších škôd a to tak, že jeho hustota v limite klesá k nule oveľa pomalšie. Keď F značí distribučnú funkciu náhodnej veličiny Z_j, potom distribučná funkcia Paretovho rozdelenia má tvar

$$F(z) = 1 - \left(\frac{\theta_j}{z + \theta_j}\right)^{\alpha},$$

kde $z > 0$, $\theta_j > 0$ a $\alpha > 0$. Takto definované Paretovo rozdelenie je označované ako tzv. *americké* a prvýkrat bolo popísané a použité v publikácii [3]. Obvyklú

definíciu, tj. $F(z) = 1 - (\frac{\theta_j}{z})^\alpha$ označuje ako *európsku* a platí, ak Z_j má európske Paretovo rozdelenie s parametrami (θ_j, α), potom $Z_j - \theta_j$ má americké Paretovo rozdelenie s rovnakými parametrami. Ďalej odvodíme strednú hodnotu daného Paretovho rozdelenia, ktorá reprezentuje očakávanú výšku škôd. Tá je potrebná pre odvodenie očakávaného počtu škôd zo vzťahu (2.2).

$$\begin{aligned}
E[Z_j] &= \int_0^\infty 1 - F(z)dz = \\
&= \int_0^\infty 1 - \left(1 - \left(\frac{\theta_j}{z+\theta_j}\right)^\alpha\right) dz = \\
&= \int_0^\infty \left(\frac{\theta_j}{z+\theta_j}\right)^\alpha dz = \theta_j^\alpha \int_0^\infty (z+\theta_j)^{-\alpha} dz = \\
&= |\text{subst. } z+\theta_j = y| = \\
&= \theta_j^\alpha \int_{\theta_j}^\infty y^{-\alpha} dy = \theta_j^\alpha \left[\frac{y^{-\alpha+1}}{-\alpha+1}\right]_{y=\theta_j}^\infty = \\
&= \theta_j^\alpha \left(0 - \frac{\theta_j^{-\alpha+1}}{-\alpha+1}\right) = \frac{\theta_j^\alpha \theta_j^{-\alpha+1}}{\alpha-1} = \\
&= \frac{\theta_j}{\alpha-1},
\end{aligned}$$

kde $\theta_j > 0$ a $\alpha > 1$, aby stredná hodnota bola konečná.

Budeme predpokladať, že pre všetky vývojové roky j bude α konštantné a parameter θ_j bude rásť s počtom rokov ubehnutých od vzniku škody. Obvykle sa predpokladá, že výška škôd je rovnaká pre všetky vývojové roky. Naša definícia Z_j je v súlade s tým, že likvidácia väčšej škody trvá dlhší čas, a teda k uhradeniu príslušných škodných nárokov dochádza pozdejšie.

Za predpokladu, že $\{Z_{j,k}\}_{k=1,2,\dots}$ sú vzájomne nezávislé a rovnako rozdelené náhodné veličiny a náhodná veličina $N_{i,j}$ na nich nezávisí, má úhrn škôd $X_{i,j}$ zložené rozdelenie. Na základe toho, že počet škôd $N_{i,j}$ má negatívne binomické rozdelenie, budeme ďalej predpokladať, že $X_{i,j}$ má zložené negatívne binomické rozdelenie, z čoho je odvodený aj názov skúmaného CNB modelu.

Poznámka (voľba rozdelení v kolektívnom modely rizika): V prípade, že by sme za rozdelenie počtu škôd $N_{i,j}$ zvolili Poissonovo rozdelenie, tak ako to urobil Meyers vo svojom ďalšom článku [6], úhrn škôd $X_{i,j}$ by mal zložené Poissonovo

rozdelenie. Model so zloženým Poissonovým rozdelením je ale vhodný iba v tom prípade, keď rozptyl počtu škôd nie je väčší ako ich priemer. Inak nie je presný a je lepšie zvoliť negatívne binomické rozdelenie.

2.4 Podmienené CNB rozdelenie

Celý CNB model, tak ako aj iné modely na výpočet rezerv na poistné plnenia, vychádza primárne z pozorovaní $x_{i,j}$ obsiahnutých vo vývojovom trojuholníku. Už vieme, že náhodná veličina $X_{i,j}$ v závislosti na iných náhodných veličinách $N_{i,j}$ a Z_j má zložené negatívne binomické rozdelenie. Pre odhad neznámych parametrov, ako aj použitie Bayesovej vety, potrebujeme ale poznať konkrétne hodnoty hustoty zloženého negatívne binomického rozdelenia, ktorú budeme značiť

$$CNB\left(x_{i,j} \mid L, \{F_j\}\right).$$

Postup, ktorý nás dovedie k výslednej hustote, má dva hlavné kroky. Najprv je nutné diskretizovať rozdelenie výšky škôd Z_j a nasledovne na diskretizované pravdepodobnosti použiť rýchlu Fourierovu transformáciu (FFT - Fast Fourier Transform). Oba kroky teraz podrobne popíšeme.

Diskretizácia rozdelenia výšky škôd

Diskretizácia je proces transformovania spojitých modelov, alebo pravdepodobnostných rozdelení na diskrétne. Takto získané premenné sú potom vhodné k použitiu ďalších numerických metód. V CNB modely pristupujeme k diskretizácii rozdelenia výšky škôd, pretože to vyžaduje FFT, ktorá pracuje len s diskrétnymi veličinami.

Prvým krokom k diskretizácii je stanovenie dĺžky diskretizačného intervalu h. Tá musí byť stanovená tak, aby celkový počet týchto intervalov (označíme ρ) zahrňoval pravdepodobný rozsah škôd. Všeobecne platí, čím jemnejšie delenie zvolíme, tým presnejší výsledok dostaneme. S ohľadom na to, že ďalším krokom bude FFT, musíme zvoliť delenie, ktoré bude mocninou dvojky. Na druhej strane zvolenie príliš jemného delenia by výrazne predĺžilo čas výpočtu Fourierovej transformácie. Konkrétna hodnota ρ bola stanovená empiricky tak, aby sa pre ňu celková doba výpočtu pohybovala v rádoch desiatok minút. Nakoniec stanovíme hodnotu k tak,

aby k–násobok h bol rovný limitu poistenia (policy limit), čo je maximálna čiastka za poistné plnenie, ktorú poisťovňa preplatí. To znamená, že v prípade, keď má poisťovňa nastavený limit poistenia na milión dolárov a poistné plnenie prekročí túto hranicu, poisťovňa už viac na plnení nevyplatí.

Konkrétne, jemnosť delenia stanovíme na $\rho = 2^{14}$ intervalov, limit poistenia na $1\,000\,000$, a teda $k = 1\,000\,000/h$. Dĺžku intervalu h stanovíme tak, aby platilo $\rho h \geq P_i$ a zároveň bola celočíselným deliteľom limitu poistenia (kvôli implementácii algoritmu diskretizácie výšky škôd), teda

- nech h' je súčet prijatého poistného za celé obdobie trvania poistenia vydelený 2^{14}, tj, $h' = \sum_{i=1}^{m} P_i/2^{14}$,
- z množiny $M = \{5, 10, 15, 20, 25, 40, 50, 100, 125, 200, 250, 500, 1000\}$, ktorá pozostáva z vybraných deliteľov limitu poistenia, vyberieme takú najmenšiu hodnotu a, že $a \geqslant h'/1\,000$,
- potom $h = 1\,000a$.

Na diskretizáciu rozdelenia výšky škôd pre každý vývojový rok použijeme metódu zachovávajúcu priemer (Mean preserving method). Táto metóda zaručuje, že diskretizované rozdelenie bude mať rovnaký priemer ako to pôvodné. Náš popis metódy vychádza z [4], str. 656. Obecne, keď chceme diskretizovať rozdelenie náhodnej veličiny X, diskrétne pravdepodobnosti budú tvaru

$$p_0 = 1 - \frac{E\left[\min(X, h)\right]}{h},$$

$$p_i = \frac{2E\left[\min(X, ih)\right] - E\left[\min\left(X, (i-1)h\right)\right] - E\left[\min\left(X, (i+1)h\right)\right]}{h}, i = 1, 2, \cdots .$$

Teraz musíme uvedené vzťahy modifikovať tak, aby sme dostali diskrétne rozdelenie náhodnej veličiny Z_j. Nech $p_{i,j}$ označuje pravdepodobnosť výskytu škody výšky $h \cdot i$ vo vývojom roku j. Namiesto stredných hodnôt $E[min(X, h)]$ budeme vychádzať z funkcie $LAS_j(h)$, ktorá vyjadruje priemernú výšku škody, keď všetky škody, ktoré boli vyplatené v j–tom roku po ich vzniku, väčšie ako h zhora obmedzíme na túto hodnotu. Takže môžme povedať, že funkcia $LAS_j(h)$ je v podstate iba prepisom vyššie uvedeného výrazu $E[min(X, h)]$. Takto definovaná diskretizačná metóda určuje pre náš model diskrétne pravdepodobnosti v štyroch nasledujúcich krokoch:

1. $p_{0,j} = 1 - \frac{LAS_j(h)}{h}$,
2. $p_{i,j} = \frac{2LAS_j(h\cdot i) - LAS_j((i-1)h) - LAS_j((i+1)h)}{h}, \quad i = 1, 2, \cdots, k-1,$
3. $p_{k,j} = 1 - \sum_{i=0}^{k-1} p_{i,j}$,
4. $p_{i,j} = 0, \quad i = k+1, \cdots, 2^{14} - 1.$

Tretí bod zaručuje, že pravdepodobnosti sa sčítajú na jednotku a štvrtý bod vyjadruje podmienku, že škody, ktoré prekročia daný limit, majú nulový výskyt.

Často sa spojité rozdelenie diskretizuje jednoduchšie a to tak, že sa zvolí delenie intervalu, na ktorom chceme diskretizovať a v jednotlivých bodoch delenia bude diskretizovaná pravdepodobnosť rovná hodnote hustoty pôvodného rozdelenia. Nakoniec sa spočíta chyba diskretizácie odvodená z toho, že pravdepodobnosti sa musia sčítať na jednotku a rozdistribuuje sa na všetky spočítané pravdepodobnosti. Takáto diskretizácia je intuitívna, avšak nezaručuje, na rozdiel od metódy popísanej vyššie, že zachová určitú vlastnosť. Potreba zachovania priemeru, či ako v našom prípade priemerných výšok škôd určených funkciou *LAS*, zaručuje, že diskretizované rozdelenie je presnejšie vzhľadom k pôvodnému rozdeleniu.

Poznámka (*LAS*): Funkcia *LAS* (*Limited Average Severity*), ako už bolo zmienené, vyjadruje akési priemerné výšky škôd. Pri obmedzení hodnotou h sa $LAS(h)$ spočíta ako priemer všetkých škôd výšky menšej ako je daná hranica a škody, ktoré ju prekračujú sa „zarovnajú“ na hodnotu tejto hranice. Do priemeru sa potom započítavajú ako škody výšky h. Štandardizované hodnoty funkcie *LAS* pre rôzne rozdelenia výšky škôd poskytuje organizácia ISO (International Organization for Standardization), ktorá tieto hodnoty vypočítava z dát o škodách, ktoré jej dodávajú poisťovne. V tretej kapitole skonštruujeme vlastnú funkciu *LAS* za pomoci náhodného výberu z rozdelenia výšky škôd.

FFT

Rýchla Fourierova transformácia je efektívny algoritmus na spočítanie rozdelenia diskrétnej náhodnej veličiny invertovaním jej charakteristickej funkcie. Vychádza pritom z klasickej diskrétnej Fourierovej transformácie, ktorú prepíše ako súčet dvoch transformácií a tým zrýchli výpočet. Keďže v našom modely vychádzame

z vektora, a teda Fourierovej transformácie dĺžky 2^{14}, je požiadavka čo najrýchlejšieho výpočtu žiadúca.

Všeobecne, pre vektor $(f_0, f_1, \cdots, f_{n-1})$ je diskrétna Fourierova transformácia zobrazenie tvaru

$$\tilde{f}_j = \sum_{k=0}^{n-1} f_k \cdot \exp\{\frac{2\pi i}{n} kj\}, \quad j = 0, \cdots, n-1,$$

kde f_k je periodická funkcia s periódou n definovaná pre celé čísla. Za rovnakých podmienok môžeme túto transformáciu prepísať do rýchlej Fourierovej transformácie rozdelením na párne a nepárne členy nasledovne

$$\begin{aligned} \tilde{f}_j &= \sum_{k=0}^{n-1} f_k \exp\{\frac{2\pi i}{n} kj\} = \\ &= \sum_{k=0}^{\frac{n}{2}-1} f_{2k} \exp\{\frac{2\pi i}{n} 2kj\} + \sum_{k=0}^{\frac{n}{2}-1} f_{2k+1} \cdot \exp\{\frac{2\pi i}{n}(2k+1)j\}. \end{aligned}$$

Takýmto spôsobom dostaneme ďalšie dve transformácie a rekurzívne znova použijeme FFT až dôjdeme k transformáciám dlžky 1. To je dôvod, prečo musíme začínať s vektorom dĺžky mocniny dvojky. Inverznú Fourierovu transformáciu a analogicky aj inverznú FFT dostaneme zo vzťahu

$$f_k = \frac{1}{n} \sum_{j=0}^{n-1} \tilde{f}_j \exp\{-\frac{2\pi i}{n} kj\}.$$

Vyššie sme uviedli ako diskretizovať rozdelenie výšky škôd, ostáva ukázať ako pomocou týchto diskrétnych pravdepodobností a FFT dospieť k podmienenému rozdeleniu úhrnu škôd $CNB(x_{i,j} \mid L, \{F_j\})$.

1. Nech $\vec{p_j}$ označuje vektor diskretizovaných pravdepodobností pre vývojový rok j vypočítaných postupom uvedeným vyššie, tj.

   $$\vec{p_j} = (p_{0,j}, p_{1,j}, \cdots, p_{2^{14}-1,j}).$$

   Spočítaním FFT pravdepodobností $\vec{p_j}$ dostaneme charakteristickú funkciu diskretizovaného rozdelenia. Výstupom bude opäť vektor dĺžky 2^{14} pre každý rok vývoju j, kde l–tá zložka výsledného vektora má tvar

   $$(\psi(\vec{p_j}))_l = \sum_{k=0}^{2^{14}-1} p_{k,j} \exp\{\frac{2\pi i}{2^{14}} kl\}, \quad l = 0, \cdots, 2^{14}-1.$$

2. Pomocou vzťahu (2.2) vypočítame očakávaný počet škôd $\lambda_{i,j}$ ako podiel stredných hodnôt nastaných škôd a výšky škôd, teda $\lambda_{i,j} = \frac{E[X_{i,j}]}{E[Z_j]}$.

3. S využitím charakteristickej funkcie diskretizovaného rozdelenia $\psi(\vec{p_j})$ a očakávaných počtov škôd $\lambda_{i,j}$ určíme vytvárajúcu funkciu rozdelenia počtu škôd P_N, ktorá v bode $\psi(\vec{p_j})$ je zároveň charakteristickou funkciou úhrnu škôd $X_{i,j}$, pretože platí vzťah

$$\psi_{X_{i,j}}(z) = P_N[\psi_{Z_j}(z)].$$

Pre negatívne binomické rozdelenie počtu škôd má náhodná veličina $N_{i,j}$ vytvárajúcu funkciu

$$\begin{aligned}
P_N[s] &= \sum_{n=0}^{\infty} s^n P(N_{i,j} = n) = \\
&= \sum_{n=0}^{\infty} s^n \binom{n + \frac{1}{c} - 1}{n} \left(\frac{c\lambda_{i,j}}{1 + c\lambda_{i,j}}\right)^n \left(\frac{1}{1 + c\lambda_{i,j}}\right)^{\frac{1}{c}} = \\
&= \sum_{n=0}^{\infty} \underbrace{\binom{n + \frac{1}{c} - 1}{n}}_{\binom{-\frac{1}{c}}{n}(-1)^n} \left(\frac{sc\lambda_{i,j}}{1 + c\lambda_{i,j}}\right)^n (1 + c\lambda_{i,j})^{-\frac{1}{c}} = \\
&= (1 + c\lambda_{i,j})^{-\frac{1}{c}} \sum_{n=0}^{\infty} \binom{-\frac{1}{c}}{n} (-1)^n \left(\frac{sc\lambda_{i,j}}{1 + c\lambda_{i,j}}\right)^n = \\
&= (1 + c\lambda_{i,j})^{-\frac{1}{c}} \sum_{n=0}^{\infty} \binom{-\frac{1}{c}}{n} \left(-\frac{s\ c\lambda_{i,j}}{1 + c\lambda_{i,j}}\right)^n = \\
&= (1 + c\lambda_{i,j})^{-\frac{1}{c}} \underbrace{\sum_{n=0}^{\infty} \binom{-\frac{1}{c}}{n} \left(-\frac{s\ c\lambda_{i,j}}{1 + c\lambda_{i,j}}\right)^n (-1)^{-\frac{1}{c}+n}}_{\text{binomická veta} \Rightarrow \left(1 - \frac{sc\lambda_{i,j}}{1+c\lambda_{i,j}}\right)^{-\frac{1}{c}}} = \\
&= (1 + c\lambda_{i,j})^{-\frac{1}{c}} \left(1 - \frac{sc\lambda_{i,j}}{1 + c\lambda_{i,j}}\right)^{-\frac{1}{c}} = \\
&= (1 + c\lambda_{i,j} - sc\lambda_{i,j})^{-\frac{1}{c}} = \\
&= (1 - c\lambda_{i,j}(s-1))^{-\frac{1}{c}}.
\end{aligned}$$

Keď do práve odvodeného výrazu za s dosadíme charakteristickú funkciu diskretizovaného rozdelenia $\psi(\vec{p_j})$, získame charakteristickú funkciu, a teda

aj Fourierovu transformáciu náhodnej veličiny $X_{i,j}$, ktorá má tvar

$$\psi(\vec{q_{i,j}}) = P_N[\psi(\vec{p_j})] = (1 - c\lambda_{i,j}(\psi(\vec{p_j}) - 1))^{-\frac{1}{c}}.$$

4. Predposledným krokom k určeniu hustoty $CNB(x_{i,j} \mid L, \{F_j\})$ je výpočet samotného vektoru $\vec{q_{i,j}}$. Ten získame použitím inverznej Fourierovej transformácie pre $\psi(\vec{q_{i,j}})$, tzn.

$$(\vec{q_{i,j}})_l = (\psi^{-1}\psi(\vec{q_{i,j}}))_l = \frac{1}{2^{14}} \sum_{k=0}^{2^{14}-1} (\vec{q_{i,j}})_k \exp\{-\frac{2\pi i}{2^{14}} kl\}$$

pre $l = 0, \cdots, 2^{14} - 1$. Algoritmus FFT potom sumu vpravo rekurzívne spočíta ako súčet párnych a nepárnych zložiek.

5. Nakoniec zvolíme k rovné takému násobku h, ktorý je najbližšie hodnote pozorovania $x_{i,j}$. Potom

$$CNB(x_{i,j} \mid L, \{F_j\}) = k\text{–ta zložka vektoru } \vec{q_{i,j}}.$$

2.5 Odhad neznámych parametrov

Ako už bolo spomínané, parametre L a $\{F_j\}$ potrebné na určenie očakávaných úhrnov škôd v Cape Cod metóde sú neznáme, a preto musia byť odhadnuté. K odhadu použijeme metódu maximálnej vierohodnosti. Táto metóda zaručuje, že odhady parametrov sú konzistentné a asymptoticky eficientné, čo znamená, že odhadujú neznáme parametre najlepším možným spôsobom.

Všeobecne, metóda maximálnej vierohodnosti pre pevnú sadu dát a daný štatistický model vyberá také hodnoty odhadovaných parametrov, pre ktoré je pravdepodobnosť tejto sady dát maximálna. Inými slovami, odhaduje parametre tak, aby maximalizovali vierohodnostnú funkciu. Pre nezávislé rovnako rozdelené náhodné veličiny $X_1, \cdots, X_n$ má vierohodnostná funkcia tvar

$$L(\{X_i\}) = \prod_{i=1}^{n} f(X_i|\theta),$$

kde θ je odhadovaný parameter rozdelenia pozorovaní $X_1, \cdots, X_n$ s hustotou f.

V CNB modely pre neznáme parametre L a $\{F_j\}$ má vierohodnostná funkcia tvar

$$L(\{x_{i,j}\}) = \prod_{i=1}^{m} \prod_{j=1}^{m+1-i} CNB(x_{i,j} \mid E[X_{i,j}]), \tag{2.3}$$

kde *CNB* označuje funkciu hustoty zloženého negatívne binomického rozdelenia a jej hodnoty dostaneme postupom popísaným v predošlej podkapitole. Maximalizácia vierohodnostnej funkcie $L(\{x_{i,j}\})$ nepriamo cez vzťah (2.1) je vlastne optimalizačnou úlohou

$$\max_{L,\{F_j\}} \prod_{i=1}^{m} \prod_{j=1}^{m+1-i} CNB(x_{i,j} \mid L, \{F_j\})$$

za podmienok

$$\sum_{j=1}^{m} F_j = 1,$$
$$0 < L < 1.$$

Daná podmienka pre vývojové faktory je odvodená z ich interpretácie ako percenta všetkých poistných udalostí vyplatených v j–tom roku po ich vzniku. Maximalizovať danú funkciu nie je v našom prípade jednoduché, pretože sa nejedná o úlohu lineárnej optimalizácie, ktorá by sa dala jednoducho previesť na úlohu nájdenia viazaných extrémov. Existuje mnoho rozličných metód na riešenie zložitejších optimalizačných úloh. Jednou z nich je aj *Nelder–Meadova metóda*.

Nelder–Meadova metóda je numerická simplexová metóda, ktorá sa obvykle používa pri riešení nelineárnych optimalizačných úloh. Napriek tomu, že táto metóda je bežnou súčasťou matematických programovacích jazykov (napr. R, Mathematica), popíšeme aspoň stručne princíp a algoritmus, ktorým sa riadi. Opäť budeme vychádzať z popisu metódy, tak ako je uvedený v [4], str. 664. Najprv však pripomeňme, že hladáme odhady parametrov L a F_j, $j = 1, \cdots, m$, teda vektor dimenzie $m + 1$.

Nech $\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_{m+2}}$ sú vektory dimenzie $m+1$ odpovedajúce vrcholom simplexu v $\mathbb{R}^{m+1}$ a f je funkcia, ktorú chceme optimalizovať. Nelder–Meadov algoritmus prebieha v nasledujúcich krokoch:

1. Zoradíme body $\mathbf{x_1}, \mathbf{x_2}, \cdots, \mathbf{x_{m+2}}$ podľa funkčných hodnôt f tak, aby platilo

$$f_1 = f(\mathbf{x_1}) \leq f_2 = f(\mathbf{x_2}) \leq \cdots \leq f_{m+2} = f(\mathbf{x_{m+2}}).$$

Potom $\mathbf{x_1}$ označuje najhorší bod *(worst point)*, $\mathbf{x_2}$ druhý najhorší bod *(second worst point)* a $\mathbf{x_{m+2}}$ najlepší bod *(best point)*.

2. Určíme $\mathbf{y_1}$ ako geometrický stred bodov $\mathbf{x_2}, \cdots, \mathbf{x_{m+2}}$ a pomocou neho si nadefinujeme ďalšie pomocné body

$$\begin{aligned}
\mathbf{y_1} &= \frac{1}{m+1}\sum_{j=2}^{m+2}\mathbf{x_j}, \\
\mathbf{y_2} &= 2\mathbf{y_1} - \mathbf{x_1}, \\
\mathbf{y_3} &= 2\mathbf{y_2} - \mathbf{x_1}, \\
\mathbf{y_4} &= \frac{(\mathbf{y_1} + \mathbf{y_2})}{2}, \\
\mathbf{y_5} &= \frac{(\mathbf{y_1} + \mathbf{x_1})}{2}.
\end{aligned}$$

3. Označíme $g_2, \cdots, g_5$ funkčné hodnoty funkcie f v bodoch $\mathbf{y_2}, \cdots, \mathbf{y_5}$, tj.

$$g_j = f(\mathbf{y_j}), j = 2, \cdots, 5.$$

   Cieľom je vymeniť najhorší bod ($\mathbf{x_1}$) za jeden z pomocných bodov a to tak, že

   - ak $f_2 < g_2 < f_{m+2}$, potom $\mathbf{x_1} \to \mathbf{y_2}$,
   - ak $g_2 \geq f_{m+2}$ a súčasne $g_3 > f_{m+2}$, potom $\mathbf{x_1} \to \mathbf{y_3}$,
   - ak $g_2 \geq f_{m+2}$ a súčasne $g_3 \leq f_{m+2}$, potom $\mathbf{x_1} \to \mathbf{y_2}$,
   - ak $f_1 < g_2 \leq f_2$, potom $\mathbf{x_1} \to \mathbf{y_4}$,
   - ak $g_2 \leq f_1$, potom $\mathbf{x_1} \to \mathbf{y_5}$.

4. Po výmene bodu $\mathbf{x_1}$ za jeden z bodov $\mathbf{y_2}, \cdots, \mathbf{y_5}$ sa opäť vrátime na prvý krok. To znamená, že zvyšné body spolu s nanovo vybratým opäť zoradíme podľa ich funkčných hodnôt a pokračujeme rovnako ako predtým ďalšími krokmi.

Tento algoritmus je potrebné opakovať, teda počítať ďalšie iterácie, pokiaľ nedostaneme $m+2$ bodov, ktoré vyhovujú podmienke „konvergencie“. Podmienka bude pritom splnená, ak priemerná smerodatná odchylka všetkých komponent vektorov

bude dostatočne malá. Ako optimálny vektor odpovedajúcich parametrov nakoniec zoberieme bod $\mathbf{x_{m+2}}$ z poslednej iterácie. Na to, aby sme dosiahli konvergenciu rýchlejšie, je potrebné zvoliť si vhodné počiatočné hodnoty. Tie sa stanovujú individuálne podľa „povahy“ úlohy, pričom by sme vždy mali brať do úvahy najmä podmienky optimalizačnej úlohy. Príkladom jedného z možných počiatočných vektorov v našom prípade sú hodnoty odhadovaných parametrov $L = \frac{1}{2}$ a $F_j = \frac{1}{m}$, $j = 1, \cdots, m$.

2.6 Bayesova veta

Bayesova veta je jedným zo základných nástrojov používaných v pravdepodobnosti a štatistike. Kľúčová je aj v našej práci, kde chceme dospieť k stochastickému modelu tvorby škodných rezerv. Vďaka Bayesovej vete sme zo získaných dát a výpočtov uvedených vyššie schopní predikovať budúce poistné plnenia.

V bayesovskej metodológii sa vychádza z apriórnych informácii o možných hodnotách parametra, tj. apriórneho rozdelenia, ktoré je získané nezávisle na pozorovaniach. Cieľom je vyjadriť aposteriórne rozdelenie parametra, ktoré berie do úvahy a reflektuje dané pozorovania. Vzťah medzi apriórnym a aposteriórnym rozdelením popisuje práve Bayesova veta. Teraz si uvedieme vetu v základnom znení pre dva náhodné javy.

Bayesova veta: Nech A a B sú náhodné javy a predpokladajme, že $P(B) > 0$. Potom platí

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)},$$

kde

- $P(A)$ je apriórna pravdepodobnosť javu A, neberie do úvahy žiadne informácie o jave B,
- $P(B)$ je apriórna pravdepodobnosť javu B,
- $P(A|B)$ je aposteriórna pravdepodobnosť javu A za podmienky javu B, je odvodená a závisí na konkrétnych hodnotách javu B,
- $P(B|A)$ je podmienená pravdepodobnosť javu B za podmienky, že nastal jav A.

Marginálna pravdepodobnosť $P(B)$ je vlastne normalizačnou konštantou, ktorá sa dá jednoducho spočítať pomocou vzťahu pre celkovú pravdepodobnosť, tj.

$$P(B) = \sum_i P(B \cap A_i) = \sum_i P(B|A_i)P(A_i),$$

kde $\{A_i\}$ je postupnosť navzájom sa vylučujúcich javov takých, že $P(\cup_i A_i) = 1$. Väčšinou ale hodnotu $P(B)$ ani nepotrebujeme spočítať a stačí nám vyjadrenie Bayesovej vety, ktorá zachová proporcionalitu pravdepodobností

$$P(A_i|B) \propto P(B|A_i)P(A_i) \quad \text{pre všetky } i.$$

Vráťme sa teraz k nášmu modelu, kde poznáme hodnoty z vývojového trojuholníka $x_{i,j}$, pre ktorý potrebujeme odhadnúť výšky budúcich plnení. Predpokladajme, že máme k dispozícii dáta za rovnako dlhé obdobie od ďalších poisťovní. Pre tieto si odhadneme neznáme parametre L a $\{F_j\}$ tak, ako sme to popísali v predošlých podkapitolách. Motiváciou, prečo odhadujeme neznáme parametre pre dáta iných poisťovní, je to, že chceme vytvoriť model vhodný aj pre menšie poisťovňe, ktoré nemôžu vychádzať len z vlastných nestabilných dát. Pre ilustráciu, nech máme k dispozícii dáta od dvadsiatich rôznych poisťovní. To znamená, že pre každú poisťovňu odhadneme parametre $\omega = \{\hat{L}, \hat{F}_1, \cdots, \hat{F}_m\}$, a teda celkom budeme mať 20 sád parametrov $\omega_1, \cdots, \omega_{20}$. Opakovaným použitím Bayesovej vety pre všetky sady parametrov nakoniec vyberieme tú, ktorá najlepšie modeluje naše pôvodné dáta, tj. maximalizuje pravdepodobnosť $\prod_{i,j} P(\omega|x_{i,j})$. Aposteriórne rozdelenie pre jednu sadu odhadnutých parametrov $\hat{L}$ a $\{\hat{F}_j\}$ za podmienky daného prvku trojuholníka $x_{i,j}$ dostaneme pritom zo vzťahu Bayesovej vety

$$P(\omega|x_{i,j}) = \frac{P(x_{i,j}|\omega) \cdot P(\omega)}{\sum\limits_{\omega \in \Omega} P(x_{i,j}|\omega) \cdot P(\omega)},$$

kde Ω je množina všetkých sád parametrov ω. Podmienená pravdepodobnosť $P(x_{i,j}|\omega)$ je určená negatívne binomickým rozdelením, ktorého hustota je vyššie označená ako $CNB(x_{i,j} \mid L, \{F_j\})$ a pravdepodobnosť $P(\omega)$ je určená apriórnym rozdelením odhadnutých parametrov.

Apriórne rozdelenie

Apriórne rozdelenie v bayesovskej štatistike je veľmi dôležité, pretože jeho voľba zásadne ovplyvní aj rozdelenie aposteriórne a tým celý výsledok, ku ktorému sa chceme dopracovať.

Pokiaľ pracujeme s nasimulovanými dátami, apriórne rozdelenie volíme s prihliadnutím k simulačnej metóde. Väčšinou ale ide o reálne dáta z praxe, a preto musíme byť pri výbere takéhoto rozdelenia opatrnejší. Typicky sa v takýchto prípadoch využívajú už získané skúsenosti z predošlých skúmaní škôd a tvorby rezerv pre rozličné poisťovne.

Z praktických dôvodov je zvykom za apriórne rozdelenie zvoliť tzv. *konjugované apriórne rozdelenie*. To znamená, že apriórne rozdelenie je aproximované jedným z vhodnej triedy rozdelení, pričom sa vychádza z pravdepodobnostného rozdelenia dát či pozorovaní. Aposteriórne rozdelenie je potom z rovnakej triedy rozdelení ako apriórne, a teda sú konjugované. Zvolenie konjugovaného rozdelenia je výhodné najmä z toho dôvodu, že výpočet aposteriórneho rozdelenia je potom jednoduchý a vyhneme sa tak zložitejším numerickým metódam, ktoré by sme inak boli nútení použiť. V tabuľke 2.2 sú uvedené konjugované rozdelenia pre konkrétne príklady rozdelení.

Rozdelenie pozorovaní	**Apriórne rozd.**	**Aposteriórne rozd.**
Poissonovo	Gamma	Gamma
Binomické	Beta	Beta
Negatívne binomické	Beta	Beta
Normálne	Normálne/Gamma	Normálne/Gamma

Tabuľka 2.2: Konjugované apriórne a aposteriórne rozdelenia

Na základe informácií z tabuľky je zrejmé, že pre CNB model, kde má podmienená pravdepodobnosť pozorovaní zložené negatívne binomické rozdelenie, sa vhodným apriórnym rozdelením javí beta rozdelenie. To je ale definované pre náhodnú veličinu a nie vektor ako to je v našom prípade, preto je potrebné zamyslieť sa nad postupom, ako dospieť k vhodnému rozdeleniu celého vektora.

Každý vektor odhadnutých parametrov ω pozostáva z odhadu škodného pomeru $\hat{L}$ a odhadov vývojových faktorov $\{\hat{F}_j\}$. Zatiaľ čo škodný pomer ovplyvňuje absolútnu veľkosť vývojového trojuholníka čo sa úhrnov škôd týka, tak vývojové faktory odzrkadľujú pomery vývoju škôd vo vnútri tohto trojuholníka. Z tohto dôvodu predpokladáme, že tieto dve zložky sú na sebe nezávislé, a preto apriórnu pravdepodobnosť $P(\omega)$ môžeme prepísať do vzťahu

$$P(\omega) = P(\hat{L}) \cdot P(\{\hat{F}_j\}).$$

Teraz už môžeme pre pravdepodobnosť škodných pomerov uvažovať beta rozdele-

nie, ktorého hustota má tvar

$$f(x) = \frac{1}{B(\alpha, \beta)} x^{\alpha-1}(1-x)^{\beta-1},$$

kde $\alpha, \beta > 0$, $B(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$ a x patrí do intervalu $(0, 1)$, čo odpovedá definícii škodného pomeru. Toto ale nie je možné predpokladať pre rozdelenie vývojových faktorov z dôvodu ako sme uviedli vyššie. Totiž pravdepodobnosť $P(\{\hat{F}_j\})$ už nie je možné rozdeliť na súčin pravdepodobností jednotlivých zložiek, pretože sú navzájom závislé a jedná sa teda o vektor. Z toho dôvodu je najlepšou voľbou rozdelenia vývojových faktorov zovšeobecnené beta rozdelenie známe pod názvom Dirichletovo. Hustota Dirichletovho rozdelenia má tvar

$$f(x_1, \cdots, x_m) = \frac{1}{B(\alpha_1, \cdots, \alpha_m)} \prod_{j=1}^{m} x_j^{\alpha_j - 1},$$

kde $\alpha_1, \cdots, \alpha_m > 0$, $B(\alpha_1, \cdots, \alpha_m) = \frac{\prod_{j=1}^{m} \Gamma(\alpha_j)}{\Gamma\left(\sum_{j=1}^{m} \alpha_j\right)}$ a $x_1, \cdots, x_m > 0$. Ďalšou podmienkou Dirichletovho rozdelenia, ktorú musí spĺňať vektor $\{x_1, \cdots, x_m\}$, je sčítanie jeho prvkov na jednotku. Toto je ale náš prípad, pretože platí $\sum_{j=1}^{m} F_j = 1$.

Zvolili sme vhodné rozdelenia pre škodné pomery a vývojové faktory, a teda výsledná hustota apriórneho rozdelenia celého vektoru ω bude súčinom hustôt týchto dvoch rozdelení. Posledné, čo ostáva určiť, sú parametre beta a Dirichletovho rozdelenia, v čom si vystačíme s metódou maximálnej vierohodnosti. K tomu využijeme sady parametrov $\omega_1, \omega_2, \cdots$ odhadnuté pre trénovacie trojuholníky. Pritom odhady škodných pomerov použijeme pri maximalizácii vierohodnostnej funkcie beta rozdelenia a odhady vývojových faktorov použijeme pri maximalizácii vierohodnostnej funkcie Dirichletovho rozdelenia a tým získame odhady parametrov $\hat{\alpha}, \hat{\beta}$ a $\hat{\alpha}_1, \cdots, \hat{\alpha}_m$. Konečne, výsledná apriórna pravdepodobnosť jednej sady odhadnutých parametrov ω bude rovná

$$P(\omega) = \frac{1}{B(\hat{\alpha}, \hat{\beta})} \hat{L}^{\hat{\alpha}-1}(1-\hat{L})^{\hat{\beta}-1} \cdot \frac{1}{B(\hat{\alpha}_1, \cdots, \hat{\alpha}_m)} \prod_{j=1}^{m} \hat{F}_j^{\hat{\alpha}_j - 1}.$$

Poznámka (predpoklad rovnakej apriórnej pravdepodobnosti): V tretej kapitole, kde budeme simulovať konkrétny prípad odhadu škodnej rezervy CNB

modelom, zavedieme zjednodušujúci predpoklad rovnakej apriórnej pravdepodobnosti pre každú sadu parametrov $\omega \in \Omega$. Tým sa nám vlastne výpočet Bayesovej vety obmedzí na vyčíslenie pravdepodobnosti úhrnu škôd za podmienky danej sady parametrov. Oproti tomu ale rozšírime množinu Ω a to tak, že odhady škodných pomerov „naškálujeme“ pomocou beta rozdelenia a každú z týchto hodnôt skombinujeme so všetkými odhadmi vývojových faktorov.

2.7 Škodná rezerva

Bayesova veta, tak ako ju máme definovanú pre náš model, spočíta aposteriórnu pravdepodobnosť parametrov ω za podmienky jedného prvku vývojového trojuholníka $x_{i,j}$, tj. $P(\omega|x_{i,j})$. My ale hľadáme najvyššiu možnú pravdepodobnosť, že odhadnutá sada parametrov najlepšie modeluje celý vývojový trojuholník, ktorého rezervu chceme odhadnúť. To znamená, že spočítame Bayesovou vetou aposteriórnu pravdepodobnosť sady ω pre každý prvok trojuholníka a tieto hodnoty medzi sebou vynásobíme, čím dostaneme vierohodnosť pre celý trojuholník

$$P(\omega|\triangle) = \prod_{i=1}^{m} \prod_{j=1}^{m+1-i} P(\omega|x_{i,j}).$$

Tento výpočet zopakujeme pre všetky sady parametrov $\omega_1, \omega_2, \cdots \in \Omega$, z pomedzi nich vyberieme tú, ktorá má najvyššiu vierohodnosť a označíme ju ω_{opt}. Pomocou tejto sady už odhadneme konečnú výšku rezervy pre poistné udalosti.

i	**Poistné**	**j**					
		1	**2**	**3**	**...**	**m-1**	**m**
1	P_1	$X_{1,1}$	$X_{1,2}$	$X_{1,3}$	$\cdots$	$X_{1,m-1}$	$X_{1,m}$
2	P_2	$X_{2,1}$	$X_{2,2}$	$X_{2,3}$	$\cdots$	$X_{2,m-1}$	$\hat{X}_{2,m}$
3	P_3	$X_{3,1}$	$X_{3,2}$	$X_{3,3}$	$\cdots$	$\hat{X}_{3,m-1}$	$\hat{X}_{3,m}$
$\vdots$	$\vdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$m-1$	P_{m-1}	$X_{m-1,1}$	$X_{m-1,2}$	$\hat{X}_{m-1,3}$	$\cdots$	$\hat{X}_{m-1,m-1}$	$\hat{X}_{m-1,m}$
m	P_m	$X_{m,1}$	$\hat{X}_{m,2}$	$\hat{X}_{m,3}$	$\cdots$	$\hat{X}_{m,m-1}$	$\hat{X}_{m,m}$

Tabuľka 2.3: Vývojový trojuholník doplnený o predikované poistné plnenia

Konečné vyčíslenie škodnej rezervy pozostáva z dvoch jednoduchých výpočtov.

1. Najprv doplníme vývojový trojuholník na štvorec, tj. vypočítame očakávané hodnoty budúcich plnení pre roky vzniku škody $i = 2, \cdots, m$ a vývojové roky $j = m + 2 - i, \cdots, m$ (viď tabuľku 2.3). Očakávané úhrny škôd pri voľbe sady parametrov ω dostaneme zo vzťahu pre metódu Cape Cod

$$\hat{X}_{i,j} = E[X_{i,j}|\omega_{opt}] = P \cdot \hat{L}(\omega_{opt}) \cdot \hat{F}_j(\omega_{opt}).$$

2. Výslednú výšku rezervy potom získame sčítaním predikovaných úhrnov škôd, tj.

$$\text{rezerva} = \sum_{i=2}^{m} \sum_{j=m+2-i}^{m} \hat{X}_{i,j}.$$

Kapitola 3

Simulácia modelu

V predchádzajúcej kapitole sme sa zaoberali popisom stochastického CNB modelu. Vieme, ako po získaní špecifických dát postupne pomocou viacerých matematických metód teoreticky dospieť k výslednej škodnej rezerve. V tejto kapitole sa budeme preto venovať praktickej aplikácii CNB modelu, hlavne jeho implementácii ako počítačového programu.

Výsledný program, ktorý počíta rezervu na budúce poistné plnenia prostredníctvom CNB modelu, je naprogramovaný v matematickom programovacom software Mathematica 8. Na rozdiel od iných programovacích jazykov má Mathematica v sebe zabudovaných mnoho funkcií a algoritmov, čím zjednodušuje a urýchľuje tvorbu vlastných programov. Zdrojové kódy programu sú uvedené v prílohe práce.

3.1 Generovanie vývojových trojuholníkov

Predtým, ako začneme s programovaním samotného modelu, potrebujeme dáta odpovedajúceho tvaru, tj. nekumulované vývojové trojuholníky. Napriek tomu, že v Českej republike pôsobí veľa neživotných poisťovní, získanie takto špecifických dát nie je jednoduché. Poisťovne totiž nemajú povinnosť poskytovať svoje historické dáta ani Českej asociácii poisťovní ani ich zverejňovať prostredníctvom výročných správ. Preto potrebujeme vytvoriť program, ktorý generuje nekumulatívne trojuholníky na základe kolektívneho modelu rizika. Nasimulované dáta, už z toho ako sú nadefinované, budú spĺňať podmienky nutné pre CNB model a ich overovanie nie je potrebné ako pri dátach reálnych.

Keďže úhrny škôd sa riadia kolektívnym modelom rizika, závisia na počte a výške škôd. Tieto veličiny majú rozdelenia s parametrami, ktoré je potrebné najprv vhodne zvoliť. V oboch prípadoch budeme vychádzať z článku [6], kde je popisovaný iný stochastický model tvorby škodných rezerv. Z neho si preberieme aj príklad vývojového trojuholníka, pre ktorý budeme chcieť odhadnúť budúce poistné plnenia.

i	**P**	**j**									
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	50 000	7 168	11 190	12 432	7 856	3 502	1 286	334	216	190	0
2	50 000	4 770	8 726	9 150	5 728	2 459	2 864	715	219	0	
3	50 000	5 821	9 467	7 741	3 736	1 402	972	720	50		
4	50 000	5 228	7 050	6 577	2 890	1 600	2 156	592			
5	50 000	4 185	6 573	5 196	2 869	3 609	1 283				
6	50 000	4 930	8 034	5 315	5 549	1 891					
7	50 000	4 936	7 357	5 817	5 278						
8	50 000	4 762	8 383	6 568							
9	50 000	5 025	8 898								
10	50 000	4 824									

Tabuľka 3.1: Vývoj poistných plnení poisťovne A (v tisícoch)

Nech nekumulatívny vývojový trojuholník daný hodnotami v tabuľke 3.1 predstavuje vývoj poistných udalostí za dobu desiatich rokov pre istú poisťovňu A. Všetky hodnoty v tabuľke sú v tisícoch dolárov, pretože vychádzajú z dát amerických poisťovní. Našou úlohou je pre poisťovňu A odhadnúť výšku budúcich poistných plnení a to prostredníctvom stochastického CNB modelu. K tomu v prvom rade potrebujeme tzv. trénovacie dáta, ktoré budeme generovať. Pri generovaní musíme postupovať tak, aby dáta veľkosťou zhruba odpovedali hodnotám v tabuľke 3.1, tzn. že v reálnom prípade by sme chceli narábať s dátami poisťovní približne rovnakého rozsahu a naviac aj jedného druhu poistenia. Pre názornosť si predstavme, že poisťovňa A poskytuje poistenia pre povinné ručenie. Potom všetky trénovacie dáta budú iba z oblasti povinného ručenia.

Na základe vyššie uvedeného budeme pri generovaní trojuholníkov predpokladať rovnaké poistné pre každý rok trvania poistenia vo výške 50 000 000, Paretovo rozdelenie pre výšky škôd a negatívne binomické rozdelenie pre počty škôd. Štruktúra samotného programu bude potom nasledovná:

- Pomocou zabudovanej funkcie `ParetoDistribution` spočítame distribúciu

Paretovho rozdelenia s parametrami α a θ_j.

- Pomocou `NegativeBinomialDistribution` spočítame distribúciu negatívne binomického rozdelenia s parametrami $\lambda_{i,j}$ a c.
- Vytvoríme funkciu `KMR`, prostredníctvom ktorej spočítame úhrn škôd kolektívnym modelom rizika pre konkrétny rok vzniku a vývojový rok tak, že zavolaním funkcie `RandomVariate` pre vyššie spočítané distribúcie dostaneme náhodný výber reprezentujúci výšku a počet škôd.
- Funkcia `GenerujT` potom opakovaným volaním `KMR` postupne pre každý rok vzniku a vývoja vygeneruje celý vývojový trojuholník.

Parametre Paretovho rozdelenia, ako už bolo zmienené, vychádzajú z rovnakého článku ako trojuholník uvedený v tabuľke 3.1, kde $\alpha = 2$ a θ_j je určená hodnotami uvedenými v tabuľke 3.2. Tie odrážajú fakt, že s pozdejšími vývojovými rokmi rastú aj výšky škôd.

Výv. rok j	**1**	**2**	**3**	**4**	**5**	**6**	**7-10**
θ_j	10 000	25 000	50 000	75 000	100 000	125 000	150 000

Tabuľka 3.2: Hodnoty parametrov θ_j

Z predošlej kapitoly vieme, že CNB model pracuje s americkým typom Paretovho rozdelenia s distribučnou funkciou

$$F(z) = 1 - \left(\frac{\theta_j}{z + \theta_j}\right)^{\alpha}.$$

Mathematica však používa európsku definíciu Paretovho rozdelenia, preto v kroku, kde robíme náhodný výber z tohto rozdelenia zavolaním `RandomVariete`, je nutné k veličinám pričítať hodnotu parametru θ_j, podľa toho, pre ktorý vývojový rok výber robíme.

Negatívne binomické rozdelenie sme definovali parametrizáciou

$$P(N_{i,j} = n) = \binom{n + \frac{1}{c} - 1}{n} \left(\frac{1}{c\lambda_{i,j} + 1}\right)^{\frac{1}{c}} \left(1 - \frac{1}{c\lambda_{i,j} + 1}\right)^{n}, \quad n = 0, 1, 2, \cdots.$$

Parameter c negatívne binomického rozdelenia stanovíme na hodnotu 0,01, rovnako ako v predošlých prípadoch vychádzame z hodnoty stanovenej v článku [6].

Nakoniec pri stanovovaní hodnôt parametru $\lambda_{i,j}$ postupujeme tak, aby výsledné vygenerované trojuholníky boli porovnateľné s trojuholníkom 3.1. Vychádzame pritom zo vzťahu (2.2), teda $\lambda_{i,j} = \frac{E[X_{i,j}]}{E[Z_j]}$, kde za $E[X_{i,j}]$ dosadíme úhrny škôd z trojuholníka 3.1. Takto určené hodnoty parametrov $\lambda_{i,j}$ sú uvedené v nasledujúcej tabuľke.

$\lambda_{i,j}$	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	254,666	158,933	88,405	37,243	12,451	3,657	0,791	0,512	0,450	0,000
2	169,600	124,103	65,066	27,155	8,742	8,146	1,694	0,519	0,000	
3	206,968	134,641	55,047	17,712	4,984	2,764	1,706	0,118		
4	185,884	100,266	46,769	13,700	5,688	6,132	1,403			
5	148,800	93,482	36,949	13,601	12,832	3,649				
6	175,288	114,261	37,795	26,306	6,723					
7	175,502	104,632	41,365	25,021						
8	169,315	119,224	46,705							
9	178,666	126,549								
10	171,520									

Tabuľka 3.3: Hodnoty parametrov $\lambda_{i,j}$

Vygenerované trojuholníky, ktoré predstavujú trénovacie dáta od 15 rôznych poisťovní sú uvedené v prílohe na konci práce, tak ako aj zdrojový kód generovacieho programu.

3.2 Funkcia LAS

Predtým, ako budeme môcť postúpiť k odhadovaniu neznámych parametrov pre vygenerované vývojové trojuholníky, musíme nadefinovať funkciu *LAS*, ktorá je nutná k diskretizácii rozdelenia výšky škôd. V predošlej kapitole sme uviedli, že hodnoty tejto funkcie vypočítava organizácia ISO, avšak takéto tabuľky nie sú voľne dostupné. Preto si funkciu *LAS* nadefinujeme sami, opäť sa bude jednať o funkciu naprogramovanú v Mathematice.

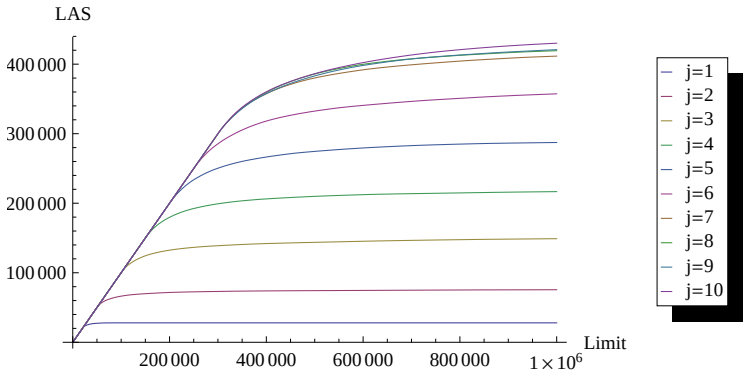
Hodnoty funkcie *LAS* predstavujú priemerné výšky škôd, ktoré postupne zhora obmedzujeme na určité hladiny. Algoritmus, ktorým definujeme funkciu *LAS* pre Paretovo rozdelenie s hodnotami parametrov uvedených vyššie, je nasledovný:

- Vytvoríme dvojrozmerné pole `ThetaAlpha`, ktoré bude obsahovať parametre Paretovho rozdelenia a určíme konštantu `maxV`, ktorá určí veľkosť vygenerovanej vzorky z Paretovho rozdelenia pre každú sadu parametrov, tj. pre každý vývojový rok.

- Definujeme funkciu `Pareto`, ktorá pomocou zabudovanej funkcie Mathematici `ParetoDistribution` spočíta distribúciu Paretovho rozdelenia pre parametre nadefinované v poli `ThetaAlpha`.

- Následne funkcia `VyskySkod` vyberie náhodnú vzorku veľkosti maxV z rozdelenia `Pareto` a funkciou `Apareto` ju prevedieme na vzorku z amerického Paretovho rozdelenia.

- Nakoniec pomocou predom spočítanej vzorky výšky škôd nadefinujeme funkciu `LAS` dvoch premenných - vývojového roku j a hranice x, na ktorú chceme saturovať výšky škôd. Funkcia `LAS` potom postupne vyberie minimá z hranice x a z každej zložky vektora predstavujúceho výšky škôd pre daný vývojový rok j. Tieto hodnoty potom posčíta a vydelí ich počtom, čím spočíta priemernú výšku škôd pre danú hranicu x.



Obr. 3.1: Graf funkcie *LAS*

Na obrázku 3.1 je uvedený graf funkcie *LAS* spočítanej práve popísaným algoritmom pre každý vývojový rok poistenia. Z grafu je viditeľné, že s ďalším vývojovým rokom rastie priemerná výška škôd tak, ako sme to predpokladali v predošlej kapitole. V prílohe je uvedený zdrojový kód programu počítajúceho hodnoty *LAS* funkcie a takisto bude ako podprogram súčasťou hlavného programu CNB model, pretože bude dynamicky volaný z podprogramu na diskretizáciu rozdelenia výšky škôd.

3.3 Optimalizácia a odhad parametrov

Zopakujme si teraz, čo je našou úlohou. Máme k dispozícii trojuholník monitorujúci vývoj poistných plnení za 10 rokov trvania poistenia v poisťovni A (viď tabuľku 3.1). Pre túto poisťovňu chceme odhadnúť výšku škodnej rezervy pomocou stochastického CNB modelu. Na tento účel máme k dispozícii dáta od ďalších 15 poisťovní, na ktorých model natrénujeme. Nakoniec vyberieme parametre najviac vyhovujúce dátam poisťovne A a pomocou týchto dopočítame škodnú rezervu.

Zatiaľ sme popísali, ako si vygenerovať potrebné vývojové trojuholníky, z ktorých budeme odhadovať neznáme parametre metódy Cape Cod. Takisto sme popísali, ako postupovať pri konštrukcii funkcie *LAS*, keď nemáme k dispozícii originálne ISO tabuľky. Teraz už pristúpime k hlavnej časti programu, a teda k algoritmom samotného CNB modelu. Hlavný program pozostáva z viacerých častí a podprogramov nazvaných zhodne s termínmi použitými v druhej kapitole, tj.:

1. diskretizácia,
2. odhad parametrov,
3. rozšírenie množiny parametrov,
4. Bayes.

Najprv musíme diskretizovať rozdelenia výšky škôd. K tomu použijeme postup popísaný v predchádzajúcej kapitole, podkapitole 2.4:

- Definujeme premenné `Poistne`, `PolicyLimit`, `rho`, a `M`.
- Definujeme pomocnú premennú `h'` ako súčet poistného podelený `rho` ($= 2^{14}$) a následne určíme dĺžku diskretizačných intervalov `h` a ich počet `k`.
- Nainicializujeme maticu `p` veľkosti $2^{14} \times 10$, ktorá bude obsahovať diskretizované pravdepodobnosti.
- `For` cyklom spočítame diskretizované pravdepodobnosti použitím vopred definovanej funkcie `LAS` postupom uvedeným na strane 13.
- Transponujeme celú maticu pravdepodobností `p` a tým získame požadovanú maticu `pT` typu 10×2^{14}, ktorá obsahuje diskretizované pravdepodobnosti pre každý vývojový rok.

```
Poistne = {50 000 000, 50 000 000, 50 000 000, 50 000 000, 50 000 000, 50 000 000,
   50 000 000, 50 000 000, 50 000 000, 50 000 000};
PolicyLimit = 1 000 000;
rozsah = 2^14;
M = {5, 10, 15, 20, 25, 40, 50, 100, 125, 200, 250, 500, 1000};
h' = Total[Poistne] / rozsah;
h = Min[Select[M, # ≥ h' / 1000 &]] * 1000;
k = PolicyLimit / h;

p = Table[0, {i, rozsah}, {j, 10}];
For[i = 1, i ≤ rozsah, i++,
  For[j = 1, j ≤ 10, j++,
   If[i == 1, p[[i, j]] = 1 - LAS[j, h] / h; Continue[]];
   If[i ≤ k, p[[i, j]] = (2 * LAS[j, h * (i - 1)] - LAS[j, (i - 2) * h] - LAS[j, i * h]) / h;
    Continue[]];
   If[i == k + 1, p[[i, j]] = 1 - Sum[p[[x, j]], {x, 1, k}]; Continue[]];
   p[[i, j]] = 0;]
 ];
pT = Transpose[p];
```

Na každý riadok matice diskretizovaných pravdepodobností `pT` zavoláme funkciu `Fourier`, ktorá prevedie rýchlu Fourierovu transformáciu a dostaneme maticu `PsiP` opäť typu 10×2^{14}. Pre budúce použitie spočítame stredné hodnoty výšky škôd `EZ`, ktoré majú americké Pareto rozdelenie. Ďalej nadefinujeme funkciu `CC`, ktorá pre daný vektor parametrov spočíta metódou Cape Cod očakávaný úhrn škôd. Odhadované parametre budú pritom zložkami jedného vektora `t` takého, že $t = \{F_1, F_2, F_3, F_4, F_5, F_6, F_7, F_8, F_9, F_{10}, L\}$. Teda v Mathematice budeme mať `CC[t]:=Table[Poistne*t[11]*t[j],{j,1,10}]`. Ďalším krokom je optimalizácia vierohodnostnej funkcie.

Na tomto mieste si treba uvedomiť, že pracujeme s vektormi dĺžky 2^{14} a narábaním s týmito vektormi chceme dospieť k odhadu neznámych parametrov pre každý jeden z pätnástich vygenerovaných trojuholníkov. Preto je namieste požiadavka čo najväčšej efektivity použitých algoritmov. Čo chceme, je maximalizovať vierohodnostnú funkciu negatívne binomického rozdelenia, tj. vzťah (2.3) za podmienky, že vývojové faktory sa sčítajú na jednotku a škodný pomer je číslo z inervalu $(0, 1)$. My k týmto podmienkam pridáme ďalšie, ktorými urýchlime proces optimalizácie.

Takže máme optimalizačnú úlohu

$$\max_t \prod_{i=1}^{m} \prod_{j=1}^{m+1-i} CNB(x_{i,j} \mid t)$$

za podmienok

$$\begin{aligned} \sum_{j=1}^{10} F_j &= 1, \\ F_1 &\leq F_2, \\ F_j &\geq F_{j+1}, \quad j = 2, \cdots, 7, \\ F_8 < 0{,}5F_7, \quad F_9 < 0{,}5F_8, \quad F_{10} < 0{,}5F_9, \\ 0 < L &< 1. \end{aligned}$$

Druhá a tretia podmienka sú vypozorované z trojuholníka 3.1, kde najvyšší úhrn škôd v priemere nastáva v druhom vývojovom roku a potom už klesá. Štvrtá podmienka pre posledné štyri vývojové faktory zase zaručí, že ich pokles bude dostatočne veľký vzhľadom k predošlým vývojovým faktorom a tým budú lepšie odpovedať realite.

Ďalšieho zrýchlenia celého výpočtu rezervy dosiahneme vhodnou voľbou počiatočného vektora pri optimalizácii vierohodnostnej funkcie Nelder–Meadovým algoritmom. Vďaka nemu môže Nelder–Meadov algoritmus stanoviť oveľa lepší štartovací simplex. Vhodnou metódou získania takýchto počiatočných hodnôt je nahradenie vierohodnostnej funkcie zloženého negatívne binomického rozdelenia vierohodnostnou funkciou rozptýleného Poissonovho rozdelenia, ktorého popis nájdeme napr. v [2]. To znamená, že prvým krokom k optimalizácii vierohodnostnej funkcie CNB modelu bude nájdenie vektora odhadov parametrov $\{F_j\}$ a L, ktoré maximalizujú logaritmickú vierohodnostnú funkciu ODP (overdispersed Poisson) rozdelenia, ktorá je tvaru

$$\begin{aligned} l_{ODP}(\{x_{i,j}\}) &= \sum_{i=1}^{10} \sum_{j=1}^{11-i} x_{i,j} \cdot (\ln E[X_{i,j}]) - E[X_{i,j}] = \\ &= \sum_{i=1}^{10} \sum_{j=1}^{11-i} x_{i,j} \cdot (\ln P_i \cdot L \cdot F_j) - P_i \cdot L \cdot F_j. \end{aligned}$$

V programe budeme postupovať nasledovne (za popisom uvedieme opäť samotný zdrojový kód):

- Definujeme vektor `xs={x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,x11}` reprezentujúci naše parametre.
- Vytvoríme premennú `podmienky`, v ktorej budú optimalizačné podmienky, tak ako sme ich nadefinovali v predchádzajúcom.
- Vytvoríme funkciu `ODPVierohodnost`, ktorá bude mať za premenné vývojový trojuholník `X` a vektor jedenástich maximalizačných bodov `s` a spočíta hodnotu funkcie `CC` v týchto bodoch. Nakoniec spočíta hodnotu logaritmickej vierohodnostnej funkcie, tak ako je definovaná vyššie.
- Nastavíme maximálny počet iterácii `MaxIterations` na 100, a tým umožníme numerickej metóde skonvergovať k dostatočne malému simplexu.
- Nazveme `maxODP` funkciu, ktorá cez zabudovanú funkciu `NMaximize` a voľby zabudovaného Nelder–Meadovho algoritmu nájde optimálne body vektora `xs`, ktoré maximalizujú funkciu `ODPVierohodnost`.
- Označíme `najODP` optimálne body `xs`.

```
xs = {x1, x2, x3, x4, x5, x6, x7, x8, x9, x10, x11};
podmienky = {x1 > 0.1, x2 > 0.2, x3 > 0.1, x4 > 0, x5 > 0, x6 > 0, x7 > 0,
   x8 > 0, x9 > 0, x10 > 0, x11 > 0.4, x11 < 0.9, x1 < x2, x2 > x3, x3 > x4,
   x4 > x5, x5 > x6, x6 > x7, x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 == 1,
   x8 < 0.5 * x7, x9 < 0.5 * x8, x10 < 0.5 * x9};
bod = {0.16801578268304668, 0.27354897383129184, 0.23909072139186907,
   0.1575745222926144, 0.07841733760303848, 0.05569662313336992,
   0.018956200943932168, 0.006102673477421544, 0.001964667948988886,
   0.0006324966944270059, 0.6147753731108555};

ODPVierohodnost[X_, s_] := Module[
   {cc},
   cc = CC[s];
   Sum[X[[i, j]] * Log[Abs[cc[[i, j]]]] - Abs[cc[[i, j]]],
    {i, 1, 10}, {j, 1, 11 - i}]];
SetOptions[NMaximize, MaxIterations → 100];
maxODP = NMaximize[{ODPVierohodnost[T, xs], podm}, xs,
   Method → {"NelderMead", InitialPoints → {bod}}];
najODP = Flatten[xs /. Rest[maxODP]];
```

Pri pohľade na zdrojový kód je vidno, že sme optimalizačné podmienky ešte upresnili a aj to, že sme zvolili počiatočný bod pre `maxODP`. Oba upresnenia boli aplikované po opakovanom použití algoritmu a overení si dodatočných podmienok.

Takže máme vektor počiatočných hodnôt a môžeme pristúpiť k maximalizácii vierohodnostnej funkcie samotného CNB rozdelenia a tým k odhadnutiu parametrov pre vygenerované trojuholníky. Ešte predtým však potrebujeme funkciu, ktorá spočíta hustotu zloženého negatívne binomického rozdelenia. Postup ako k hustote dospejeme je podrobne popísaný v podkapitole 2.4 a podľa neho je skonštruovaná v našom programe funkcia `CNB`. Tá ako vstup dostane jeden prvok z trojuholníka $x_{i,j}$, očakávaný počet škôd $\lambda_{i,j}$ a vopred spočítanú Fourierovu transformáciu pre j–ty rok $psiP_j$. Označíme `psiQ` charakteristickú funkciu úhrnu škôd vyjadrenú vzťahom $\psi(\vec{q_{i,j}}) = \left(1 - c\lambda_{i,j}(\sqrt{2^{14}}\psi(\vec{p_j}) - 1)\right)^{-\frac{1}{c}}$, kde Fourierovu transformáciu diskretizovaných pravdepodobností násobíme odmocninou z rho, pretože Mathematica Fourierovu transformáciu definuje normovanú o túto odmocninu. V ďalšom kroku funkcia `CNB` spočíta pomocou zabudovanej funkcie `InverseFourier` inverznú Fourierovu transformáciu `Q` a z rovnakého dôvodu ju tentoraz podelí odmocninou z 2^{14}. Nakoniec označíme `i` celú časť z podielu $\frac{x_{i,j}}{h}$ a výstupom bude odhad hustoty Q_i pre daný prvok trojuholníka.

```
CNB[x_, lambda_, psip_] := Module[
   {psiQ, Q, i, c, res},
   psiQ = (1 - 0.01 * lambda * (Sqrt[rho] * psip - 1)) ^ (-100);
   Q = InverseFourier[psiQ] / Sqrt[rho];
   i = IntegerPart[x / h] + 1;
   Q[[i]]];
```

Ďalšie funkcie `CNBVierohodnost` a `maxCNB` sú takmer totožné z vyššie uvedenými funkciami `ODPVierohodnost` a `maxODP`. Vďaka týmto funkciám dospejeme k odhadu neznámych parametrov Cape Cod metódy podobným postupom ako v prípade rozptýleného Poissonovho rozdelenia:

- Vytvoríme funkciu `CNBVierohodnost`, ktorá dostane za premenné vývojový trojuholník `X` a vektor maximalizačných bodov `s` a spočíta hodnotu funkcie `CC` v týchto bodoch. Spočíta hodnoty očakávaného počtu škôd `lambda` ako podiel `CC` a strednej hodnoty výšky škôd `EZ`. Nakoniec spočíta hodnotu vierohodnostnej funkcie ako násobky predom definovanej `CNB` funkcie.

- Nazveme `maxCNB` funkciu, ktorá cez zabudovanú funkciu `NMaximize` a voľbou `najODP` ako počiatočných bodov Nelder–Meadovho algoritmu nájde optimálne body vektoru `xs`, ktoré maximalizujú funkciu `CNBVierohodnost` za rovnakých podmienok ako vyššie. Vďaka vhodne zvolenému počiatočnému bodu môžeme počet iterácií znížiť na 30.

- Označíme `najCNB` optimálne body `xs`.

```
CNBVierohodnost[X_, s_?(VectorQ[#, NumericQ] &)] := Module[
   {cc, lambda}, cc = CC[s];
   lambda = Table[cc[[i, j]] / EZ[[j]], {i, 10}, {j, 1, 11 - i}];
   Product[CNB[X[[i, j]], lambda[[i, j]], PhiP[[j]]],
    {i, 10}, {j, 1, 11 - i}]];

SetOptions[NMaximize, MaxIterations → 30];
maxCNB = NMaximize[{CNBVierohodnost[T, xs], podm},
   xs, Method → {"NelderMead", InitialPoints → {najODP}}];
najCNB = Flatten[xs /. Rest[maxCNB]];
```

Poznámka (výpočtový problém v Mathematice): Pri spustení vyššie uvedenej funkcie `CNBVierohodnost` sa opakovane stávalo, že program hlásil behovú chybu v mieste výpočtu inverznej Fourierovej transformácie. Konkrétne sa jednalo o hlášku „InverseFourier: Argument is not a non-empty list or rectangular array of numeric quantities“ a nebol schopný dokončiť výpočet. Tento problém nastal, pretože vo vnútri funkcie `CNBVierohodnost` sa volala ďalšia funkcia `CNB`, ktorá sa opäť vnorovala a počítala inverzné Fourierove transformácie veľkých vektorov. Takéto zložité symbolické úpravy Mathematica nedokázala vykonať, a preto ju bolo nutné prinútiť, aby postupne robila numerické výpočty. Toto sme zaistili príkazom v mieste definície vstupných premenných `?(VectorQ[x, NumericQ])`.

Doteraz sme popísali algoritmus a časť programu, ktorý keď dostane na vstupe vývojový trojuholník, tak na výstupe dodá odhady $\hat{L}$ a $\{\hat{F}_j\}$ neznámych parametrov. V našom príklade chceme odhadnuť rezervu poisťovne A za pomoci odhadov neznámych parametrov od 15 rôznych poisťovní a to tak, že cez Bayesovu vetu nájdeme takú sadu parametrov, ktoré poskytnú najlepší požadovaný odhad rezervy.

3.4 Bayesovský odhad škodnej rezervy

Opakovaným použitím vyššie popísaného programu sme dostali 15 sád parametrov a uložili ich do premennej `Parametre`. Každá sada obsahuje 11 odhadov parametrov v poradí $\{\hat{F}_1, \hat{F}_2, \hat{F}_3, \hat{F}_4, \hat{F}_5, \hat{F}_6, \hat{F}_7, \hat{F}_8, \hat{F}_9, \hat{F}_{10}, \hat{L}\}$. K tomu, aby výsledný odhad rezervy bol čo najpresnejší, je lepšie mať k dispozícii čo najviac sád takýchto parametrov. Preto zoberieme odhady škodného pomeru zo všetkých sád parametrov a na ich základe odhadneme parametre beta rozdelenia ako konjugovaného rozdelenia pomocou maximálnej vierohodnosti. Určíme strednú hodnotu tohto rozdelenia a posúvaním sa smerom nahor aj nadol od tejto strednej hodnoty o krok dĺžky 0,03 získame ďalšie hodnoty $\hat{L}$ z beta rozdelenia. Takýmto spôsobom získame 9 rôznych odhadov parametru L a každú z pätnástich sád odhadov vývojových faktorov $\{F_j\}$ nezávisle skombinujeme s každým z týchto deviatich odhadov škodného pomeru. To znamená, že nakoniec budeme mať 135 sád parametrov vyššie označených ako ω. Funkcia, ktorá rozšíri množinu Ω o ďalšie sady parametrov práve popísaným postupom je v programe nazvaná `Rozsir`.

```
Rozsir[par_] := Module[
   {Ls, maxBeta, najBeta, EL, KROK},
   Ls = Map[#[[11]] &, par];
   maxBeta = NMaximize[{Likelihood[BetaDistribution[a, b], Ls], a > 0, b > 0,
      a < 100, b < 100}, {a, b}];
   najBeta = Flatten[{a, b} /. maxBeta[[2]]];
   EL = Mean[BetaDistribution[najBeta[[1]], najBeta[[2]]]];
   KROK = 0.03;
   Ls = Table[EL + i * KROK, {i, -4, 4}];
   Flatten[Table[Flatten[{Take[par[[i]], 10], Ls[[j]]}],
     {i, Length[par]}, {j, 9}], 1]];
```

V ďalšom kroku zadefinujeme premennú `Omega`, do ktorej uložíme vygenerované sady parametrov pomocou funkcie `Rozsir`, tj. `Omega=Rozsir[Parametre]`. Budeme ďalej intuitívne predpokladať, že každá sada parametrov $\omega \in \Omega$ má rovnakú apriórnu pravdepodobnosť (viď poznámku na strane 22). To znamená, že výpočet aposteriórnej pravdepodobnosti ω cez Bayesovu vetu sa nám zjednoduší, pretože

$$P(\omega|x_{i,j}) = \frac{P(x_{i,j}|\omega) \cdot P(\omega)}{\sum\limits_{\omega\in\Omega} P(x_{i,j}|\omega) \cdot P(\omega)} = \frac{P(\omega) \cdot P(x_{i,j}|\omega)}{P(\omega) \cdot \sum\limits_{\omega\in\Omega} P(x_{i,j}|\omega)} = \frac{P(x_{i,j}|\omega)}{\sum\limits_{\omega\in\Omega} P(x_{i,j}|\omega)}.$$

Vyjadrené proporcionálne máme

$$P(\omega|x_{i,j}) \propto P(x_{i,j}|\omega).$$

Pravdepodobnosť $P(x_{i,j}|\omega)$ vieme spočítať pomocou funkcie `CNB`. Nakoniec vyberieme tú sadu parametrov ω, ktorá najlepšie modeluje naše dáta poisťovne A. Inak povedané, pre každú sadu parametrov spočítame vierohodnosť už definovanú funkciou `CNBVierohodnost` a vyberieme tú s najväčšou hodnotou. Štruktúra týchto krokov vo finálnom programe bude nasledovná:

- Pomocou funkcie `CNB` vypočítame hustotu pre daný prvok vývojového trojuholníka a funkcia `CNBVierohodnost` jej opakovaným volaním spočíta vierohodnosť vývojového trojuholníka za podmienky danej sady parametrov ω.
- Skonštruujeme novú funkciu `Bayes`, ktorá dostane za premenné trojuholník a celú množinu parametrov Ω a zavolaním funkcie `CNBVierohodnost` spočíta vierohodnosť trojuholníka pre každú jednu sadu parametrov ω.
- Ďalej definujeme funkciu `maxBy`, ktorá porovnávaním zistí, pre ktorú sadu parametrov je vierohodnosť najväčšia a tieto parametre uloží do premennej `najParametre`.

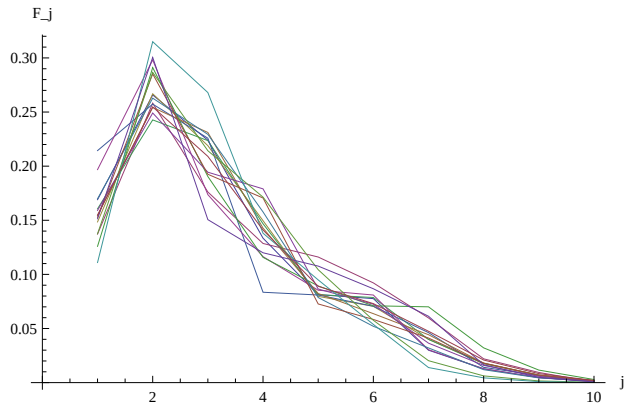
```
Bayes[X_, Omega_] := Map[{CNBVierohodnost[X, #], #} &, Omega];
maxBy[list_, n_] := With[{s = list[[All, n]]}, Pick[list, s, Max[s]]];

Omega = Rozsir[Parametre];
likelihood = Bayes[Trojuholnik, Omega];
najParametre = maxBy[likelihood, 1][[1]][[2]];
```

Teraz, keď už poznáme parametre, ktoré najlepšie modelujú naše dáta, ostáva spočítať rezervu. Funkcia `CC`, ktorá na vstupe dostane práve zvolené parametre, spočíta očakávaný úhrn škôd a teda odhad budúcich poistných plnení $\hat{X}_{i,j}$ pre $i = 2, \cdots, 10$ a $j = 12 - i, \cdots, 10$. Konečne, sčítaním týchto hodnôt dostaneme požadovanú výšku rezervy, ktorá je uložená v premennej `RezervaCelkom`. Celý zdrojový kód CNB modelu je uvedený v prílohe práce.

3.5 Porovnanie výsledkov CNB modelu a metódy Chain–Ladder

V práci sme sa najprv podrobne zaoberali teóriou stochastického CNB modelu, následne sme popísali, ako sme postupovali pri jeho implementácii ako počítačového programu a teraz nám ostáva doplniť výsledky, ktoré vzišli z použitia nášho modelu a programu pre konkrétne dáta.

Pre pätnásť vygenerovaných trojuholníkov program najprv odhadol sadu neznámych parametrov $\{F_1, \cdots, F_{10}, L\}$. Vývojové faktory $\{F_j\}$ boli odhadnuté tak, aby spĺňali podmienky zo strany 32, o čom sa môžeme presvedčiť z grafu na obrázku 3.2, kde sú tieto odhady zobrazené.



Obr. 3.2: Vývojové faktory

Odhady škodného pomeru L boli rozškálované pomocou beta rozdelenia, čím sme získali deväť „normovaných“ odhadov $\hat{L}$. Ďalej náš program každú sadu parametrov $\{\hat{F}_j\}$ skombinoval s každým normovaným odhadom $\hat{L}$ a tým sme získali 135 sád parametrov, z ktorých program vybral tú, ktorá najlepšie modelovala zadaný trojuholník poisťovne A. Pre vývojový trojuholník 3.1 sme dostali nasledujúce odhady vývojových faktorov

$$\hat{F}_1 = 0{,}196806, \quad \hat{F}_2 = 0{,}298146, \quad \hat{F}_3 = 0{,}173677, \quad \hat{F}_4 = 0{,}116402,$$
$$\hat{F}_5 = 0{,}0853722, \quad \hat{F}_6 = 0{,}0809338, \quad \hat{F}_7 = 0{,}0303917, \quad \hat{F}_8 = 0{,}012804,$$
$$\hat{F}_9 = 0{,}00463342, \quad \hat{F}_{10} = 0{,}000833472$$

a škodný pomer bol odhadnutý na

$$\hat{L} = 0{,}633632.$$

S vhodnými odhadmi parametrov ďalej program vypočítal odhady budúcich poistných plnení cez vzťah (2.1), tj.

$$\hat{X}_{i,j} = P \cdot \hat{L} \cdot \hat{F}_j, \quad i = 2, \cdots, 10,\ j = 12 - i, \cdots, 10$$

a v tabuľke 3.4 je doplnený trojuholník poisťovne A o budúce poistné plnenia.

i	**j**									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	7 168	11 190	12 432	7 856	3 502	1 286	334	216	190	0
2	4 770	8 726	9 150	5 728	2 459	2 864	715	219	0	26,405
3	5 821	9 467	7 741	3 736	1 402	972	720	50	146,794	26,405
4	5 228	7 050	6 577	2 890	1 600	2 156	592	405,650	146,794	26,405
5	4 185	6 573	5 196	2 869	3 609	1 283	962,859	405,650	146,794	26,405
6	4 930	8 034	5 315	5 549	1 891	2 564,11	962,859	405,650	146,794	26,405
7	4 936	7 357	5 817	5 278	2 704,73	2 564,11	962,859	405,650	146,794	26,405
8	4 762	8 383	6 568	3 687,81	2 704,73	2 564,11	962,859	405,650	146,794	26,405
9	5 025	8 898	5 502,37	3 687,81	2 704,73	2 564,11	962,859	405,650	146,794	26,405
10	4 824	9 445,73	5 502,37	3 687,81	2 704,73	2 564,11	962,859	405,650	146,794	26,405

Tabuľka 3.4: Odhad budúcich p. plnení poisťovne A (v tisícoch)

Zo vzťahu pre model Cape Cod, ktorým budúce poistné plnenia modelujeme, je zrejmé, že pri rovnakom poistnom pre každý rok vzniku škôd závisí tento vzťah len od indexu j, teda od vývojového roku. Z tohto dôvodu majú odhadnuté poistné plnenia v stĺpcoch rovnakú výšku pre každý rok vzniku škody.

Nakoniec, po sčítaní všetkých odhadov budúcich plnení, výsledná škodná rezerva odhadnutá stochastickým CNB modelom pre poisťovňu A vyšla vo výške

$$\text{rezerva} = 65\,182\,071.$$

Poznámka (odhad rezervy pre „diagonálu“): Predstavme si, že chceme odhadnúť rezervu s tým, že máme k dispozícii dáta o poistných plneniach vyplatených v poslednom roku poistenia. To znamená, že poznáme hodnoty úhrnov škôd z diagonály vývojového trojuholníka a nemôžeme pre odhad rezervy použiť žiadnu z klasických metód. Výhodou CNB modelu je, že aj v takomto prípade je možné pomocou neho odhadnúť výšku rezervy, aj keď výsledok môže byť menej presný. Pre porovnanie sme zobrali do úvahy iba hodnoty z diagonály trojuholníka 3.1. Výsledná rezerva nám vyšla zhruba o 3% nižšia oproti pôvodnej úlohe vo výške 63 124 200. Môžeme teda povedať, že v porovnaní sa oba odhady od seba líšia iba minimálne.

Chain–Ladder

Pre porovnanie spočítame škodnú rezervu pre vývojový trojuholník poisťovne A tiež metódou Chain–Ladder. V jednej z poznámok v druhej kapitole sme už metódu Chain–Ladder spomenuli, konkrétne sme popísali možnosť odhadnutia neznámych parametrov modelu Cape Cod pomocou tejto metódy. Teraz metódu Chain–Ladder popíšeme podrobnejšie.

Metóda Chain–Ladder pre odhad škodnej rezervy sa používa pre kumulatívne vývojové trojuholníky. Preto ako prvý krok musíme previesť hodnoty z trojuholníka 3.1 na kumulatívne vzťahom $Y_{i,j} = \sum_{k=1}^{j} X_{i,k}$, kde $X_{i,j}$ sú prírastkové úhrny škôd. Prevedený kumulatívny trojuholník je uvedený v tabuľke 3.5.

i	**j**									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	7 168	18 358	38 646	42 148	42 148	43 434	43 768	43 984	44 174	44 174
2	4 770	13 496	22 646	28 374	30 833	33 697	34 412	34 631	34 631	
3	5 821	15 288	23 029	26 765	28 167	29 139	29 859	29 909		
4	5 228	12 278	18 855	21 745	23 345	25 501	26 093			
5	4 185	10 758	15 954	18 823	22 432	23 715				
6	4 930	12 964	18 279	23 828	25 719					
7	4 936	12 293	18 110	23 388						
8	4 762	13 145	19 713							
9	5 025	13 923								
10	4 824									

Tabuľka 3.5: Kumulatívny vývojový trojuholník (v tisícoch)

Ako ďalšie vypočítame vývojové koeficienty f_j zo vzťahu

$$f_j = \frac{\sum_{i=1}^{10-j} Y_{i,j+1}}{\sum_{i=1}^{10-j} Y_{i,j}}, \quad j = 1, \cdots, 9$$

a následne spočítame odhady nakumulovaných škôd pre posledný desiaty rok poistenia v každom roku vzniku škôd cez vzťah

$$\hat{Y}_{i,10} = Y_{i,11-i} \cdot f_{11-i} \cdots f_9, \quad i = 2, \cdots, 10.$$

Výsledná rezerva bude súčet týchto odhadov, od ktorých odpočítame kumulatívne úhrny z diagonály, tj.

$$\text{rezerva} = \sum_{i=2}^{10} (\hat{Y}_{i,10} - Y_{i,11-i}).$$

Tak ako pre CNB model, tak aj v prípade metódy Chain–Ladder sme výpočty realizovali cez program v Mathematice 8 a zdrojový kód je opäť uvedený v prílohe. Vývojové koeficienty v metóde Chain–Ladder pre vývojový trojuholník 3.5 nám vyšli

$$f_1 = 2{,}61619, \quad f_2 = 1{,}5415, \quad f_3 = 1{,}22962, \quad f_4 = 1{,}09143,$$
$$f_5 = 1{,}05827, \quad f_6 = 1{,}01792, \quad f_7 = 1{,}00449, \quad f_8 = 1{,}00242, \quad f_9 = 1.$$

Následne kumulatívne úhrny škôd pre posledný desiaty rok poistenia vyšli

$$\hat{Y}_{2,10} = 34\,631\,000, \quad \hat{Y}_{3,10} = 29\,981\,300, \quad \hat{Y}_{4,10} = 26\,273\,500,$$
$$\hat{Y}_{5,10} = 24\,306\,900, \quad \hat{Y}_{6,10} = 27\,896\,900, \quad \hat{Y}_{7,10} = 27\,688\,000,$$
$$\hat{Y}_{8,10} = 28\,696\,000, \quad \hat{Y}_{9,10} = 31\,242\,500, \quad \hat{Y}_{10,10} = 28\,319\,700.$$

Nakoniec spočítaná rezerva pre budúce poistné plnenia metódou Chain–Ladder vyšla vo výške

$$\text{rezerva} = 57\,120\,700.$$

Porovnaním výsledných škodných rezerv odhadnutých pomocou CNB modelu a metódy Chain–Ladder zistíme, že nami predstavený CNB model prevyšuje rezervu metódy Chain–Ladder o 8 061 371, teda asi o 12%. Aby sme však neporovnávali len výsledné rezervy, porovnáme ešte celkové kumulatívne úhrny škôd oboch metód. To znamená, že dopočítame kumulatívne úhrny škôd v doplnenom trojuholníku CNB modelu tak, že sčítame hodnoty na každom riadku. V tabuľke 3.6 sú tieto hodnoty uvedené spolu s kumulatívnymi úhrnmi vypočítanými metódou Chain–Ladder.

	CNB model	Chain–Ladder
$\hat{Y}_{2,10}$	34 657 405	34 631 000
$\hat{Y}_{3,10}$	30 082 199	29 981 300
$\hat{Y}_{4,10}$	26 671 849	26 273 500
$\hat{Y}_{5,10}$	25 256 708	24 306 900
$\hat{Y}_{6,10}$	29 824 818	27 896 900
$\hat{Y}_{7,10}$	30 198 548	27 688 000
$\hat{Y}_{8,10}$	30 211 358	28 696 000
$\hat{Y}_{9,10}$	29 923 728	31 242 500
$\hat{Y}_{10,10}$	30 270 458	28 319 700

Tabuľka 3.6: Celkové kumulatívne úhrny škôd

Vzhľadom k tomu, s akými veľkými číslami pracujeme, môžeme po porovnaní výšok rezerv a celkových kumulatívnych úhrnov škôd tvrdiť, že obe metódy sú dostatočne porovnateľné.

Poznámka (časová náročnosť výpočtov): Už sme spomínali, že CNB model je náročný hlavne vzhľadom k času, ktorý je potrebný na výpočet odhadu parametrov. To je spôsobené hlavne opakovaným volaním Fourierovej transformácie dĺžky 2^{14} vo vnútri funkcie, ktorá počíta hustotu CNB rozdelenia v každej iterácii optimalizácie. V našom prípade sme program spúšťali na počítači s procesorom Intel i5 2500K a s operačnou pamäťou 8 GB. Jedná sa teda o veľmi výkonný počítač a aj napriek tomu trvalo asi 50 minút od spustenia programu CNBmodel, kým sa dopracoval k vyčísleniu rezervy. Oproti tomu doba behu programu Chain–Ladder bola minimálna, keďže sa jedná o priamočiary výpočet. Presný čas trvania jednotlivých výpočtov sme zmerali pomocou integrovanej funkcie *Timing* a časy týchto výpočtov v sekundách sú uvedené v nasledujúcej tabuľke.

Program	**Čas [s]**
CNBmodel	
Diskretizácia	0,609
Odhad parametrov	3038,48 (50,64 min)
Množiny parametrov	$< 0{,}001$
Bayes	14,52
Chain–Ladder	$< 0{,}001$

Tabuľka 3.7: Čas výpočtov v programe CNBmodel

Kapitola 4

Záver

Poisťovníctvo sa neustále vyvíja. V priebehu rokov sa objavovali nové prístupy nazerania na problematiku týkajúcu sa tvorby škodných rezerv a aktuári sa stále pokúšajú prísť na spôsob, ako zlepšiť a zdokonaliť tento proces. Cieľom našej práce bolo naštudovať stochastický CNB model využívajúci nové postupy, podrobne tieto postupy popísať a v neposlednom rade otestovať použiteľnosť modelu jeho aplikovaním ako počítačového programu.

Začali sme predstavením základných vlastností modelu. CNB model je stochastická metóda na odhad škodnej rezervy, tj. rezervy na poistné plnenia, ktorá vychádza zo základného predpokladu, že úhrny škôd sa riadia kolektívnym modelom rizika. Ďalej sme predpokladali, že máme k dispozícii mimo nášho vývojového trojuholníka, pre ktorý sme chceli rezervu odhadnúť, aj dáta o vývoji poistných plnení od iných poisťovní, tzv. trénovacie trojuholníky. Pre tieto sa budúce poistné plnenia modelujú metódou Cape Cod, ktorej neznáme parametre sme museli odhadnúť. Výsledná rezerva bola nakoniec odhadnutá za pomoci Bayesovej vety, ktorá spomedzi odhadnutých parametrov vybrala tie, ktoré najlepšie modelovali budúce poistné plnenia nášho vývojového trojuholníka. Celý model sa dá stručne popísať v nasledujúcich bodoch:

1. Trénovacie trojuholníky modelované metódou Cape Cod ($X_{i,j} = P_i \cdot L \cdot F_j$).
2. Výpočet hustoty úhrnu škôd CNB.
    - Diskretizácia rozdelenia výšky škôd.
    - FFT.

3. Odhad neznámych parametrov $L, \{F_j\}$ metódou maximálnej vierohodnosti ($\max \prod CNB$).
    - Maximalizácia metódou Nelder–Mead.
4. Použitie Bayesovej vety na nájdenie najvhodnejších parametrov $\hat{L}, \{\hat{F}_j\}$ pre daný vývojový trojuholník.
5. Odhad budúcich úhrnov škôd ($\hat{X}_{i,j} = P_i \cdot \hat{L} \cdot \hat{F}_j$).
6. rezerva $= \sum \hat{X}_{i,j}$.

V tretej kapitole sme sa zaoberali implementáciou a simuláciou CNB modelu, k čomu sme použili matematicko-programovací software Mathematicu 8. Vzhľadom k tomu, že sme nemali k dispozícii reálne dáta, tak sme najprv museli vytvoriť program, ktorý dokáže vygenerovať vývojové trojuholníky požadovaných vlastností a spĺňujúce predpoklady modelu. Takto sme získali 15 trénovacích trojuholníkov, za pomoci ktorých sme podľa vyššie uvedenej schémy dokázali odhadnúť výšku škodnej rezervy pre jeden konkrétny trojuholník prebratý z článku [6]. Podrobne sme popísali, ako sme postupovali pri programovaní CNB modelu a ktoré integrované funkcie sme pri tom využili.

Keďže, ako už bolo spomínané, sme nemali k dispozícii žiadne reálne historické ani súčasné dáta, pre kontrolu sme odhadli škodnú rezervu pre rovnaký trojuholník ešte raz a to metódou Chain–Ladder. Stochastický CNB model a Chain–Ladder sú rozdielne metódy, a preto sa nemožno čudovať, že odhady škodných rezerv vyšli odlišne. Rádovo ale ide o zrovnateľne vysoké odhady, o čom vypovedá aj tabuľka 3.6, kde sú porovnané kumulatívne odhady škôd v oboch prípadoch. Zatiaľ čo CNB metóda odhadla výšku rezervy na 65 182 100, výsledným odhadom metódy Chain–Ladder bola rezerva výšky 57 120 700.

Stochastický CNB model je sofistikovaným modelom tvorby škodných rezerv, ktorý oproti iným modelom vďaka využitiu bayesovskej metodológie dokáže zúročiť skúsenosti viacerých poisťovní a tým čo najlepšie predpovedať budúci vývoj poistných plnení. Hlavne pre tento dôvod je CNB model výhodný najmä pre menšie alebo nové poisťovne a poisťovacie spoločnosti, ktoré nedokážu dostatočne vierohodne odhadnúť výšku svojej škodnej rezervy len na základe vlastných historických dát.

Literatúra

[1] Cipra, T. (2002): *Kapitálová přiměřenost ve financích a solventnost v pojišťovnictví*, Ekopress, Praha, ISBN 80-86119-54-8.

[2] Clark, D. R. (2003): *LDF Curve Fitting and Stochastic Loss Reserving: A Maximum Likelihood Approach*, Casualty Actuarial Society Forum: 41-92.

[3] Hog, R. V. a Klugman, S. A. (1984): *Loss Distributions*, John Wiley & Sons, New York.

[4] Klugman, S. A., Panjer, H.H. a Willmot, G.E. (2004): *Loss Models: From Data to Decisions*, John Wiley & Sons, New Jersey.

[5] Meyers, G. G. (2007): *Estimating Predictive Distributions for Loss Reserve Models*, Variance 1(2): 248-272.

[6] Meyers, G. G. (2009): *Stochastic Loss Reserving with the Collective Risk Model*, Variance 3(2): 239-269.

[7] Meyers, G. G. (2006): *The Common Shock Model for Corelated Insurance Losses*, Casualty Actuarial Society Forum: 231-254.

Prílohy

Zoznam príloh

- Vygenerované trojuholníky
- Generátor vývojových trojuholníkov (zdrojový kód)
- Funkcia LAS (zdrojový kód)
- CNB model (zdrojový kód)
- Chain–Ladder (zdrojový kód)

Vygenerovane trojuholniky

9.7603*^6 11.0517*^6 11.7904*^6 7.1061*^6 3.5724*^6 1.0360*^6 461674 317034 313754 0
4.4862*^6 9.3088*^6 7.9771*^6 6.2088*^6 3.8526*^6 3.7464*^6 756295 795765 309153
5.2045*^6 9.8799*^6 7.9331*^6 2.2476*^6 1.2269*^6 846779 818015 329378
5.6104*^6 7.4663*^6 8.9508*^6 1.9377*^6 1.4706*^6 3.4541*^6 637426
3.7193*^6 7.5649*^6 3.9442*^6 2.0756*^6 3.5370*^6 3.8437*^6
5.8633*^6 7.2648*^6 4.5230*^6 5.1043*^6 2.2268*^6
5.2056*^6 6.8842*^6 6.8267*^6 6.5033*^6
6.3812*^6 9.2743*^6 8.1639*^6
5.8474*^6 10.8866*^6
5.2079*^6

8.4983*^6 10.6396*^6 12.1770*^6 7.1906*^6 4.1121*^6 261431 711704 1.3175*^6 778642 0
4.4045*^6 7.8787*^6 14.3498*^6 6.2314*^6 2.8919*^6 2.1884*^6 1.3631*^6 412715 590617
6.7700*^6 10.7492*^6 8.7951*^6 3.2706*^6 1.5916*^6 251446 802555 319692
6.2096*^6 6.3990*^6 5.8957*^6 2.4024*^6 1.0728*^6 1.8766*^6 1.6362*^6
4.3426*^6 5.7817*^6 4.8803*^6 3.2673*^6 6.7188*^6 1.7369*^6
5.8761*^6 6.3309*^6 8.1155*^6 5.6471*^6 1.1888*^6
5.2492*^6 7.6680*^6 7.8647*^6 6.0971*^6
5.9294*^6 7.7613*^6 6.4932*^6
4.5161*^6 9.8570*^6
5.7193*^6

7.6027*^6 10.0687*^6 13.7670*^6 8.8126*^6 2.9579*^6 1.2755*^6 673328 789829 325744 0
5.3466*^6 9.5069*^6 12.2567*^6 5.8045*^6 4.1873*^6 4.1578*^6 2.9834*^6 318286 406456
6.1778*^6 10.1466*^6 7.9361*^6 3.3530*^6 2.1038*^6 1.7096*^6 1.0397*^6 746659
6.6279*^6 8.7280*^6 7.3393*^6 4.8371*^6 1.0020*^6 2.6548*^6 693862
3.7601*^6 5.8145*^6 7.0552*^6 3.8488*^6 6.4824*^6 3.3413*^6
5.1661*^6 7.8170*^6 7.8137*^6 7.5696*^6 865329
6.0283*^6 8.1591*^6 3.6948*^6 5.5763*^6
5.6717*^6 10.7198*^6 6.6239*^6
5.1616*^6 19.5291*^6
5.1849*^6

8.0943*^6 11.7497*^6 11.5213*^6 4.6265*^6 2.6682*^6 1.3098*^6 640157 725666 332770 0
6.0632*^6 7.4367*^6 8.6160*^6 7.0638*^6 2.6347*^6 6.8820*^6 2.2867*^6 324419 1.0290*^6
5.8386*^6 8.5182*^6 8.0328*^6 5.2434*^6 1.2430*^6 822272 1.1358*^6 307990
4.7238*^6 7.6625*^6 6.2406*^6 4.0164*^6 2.4365*^6 3.7766*^6 790540
4.6637*^6 7.3679*^6 7.0922*^6 3.7255*^6 3.3350*^6 2.3889*^6
5.2707*^6 9.2533*^6 6.9190*^6 4.9742*^6 2.9777*^6
5.5022*^6 8.1099*^6 6.5039*^6 6.5061*^6
6.0999*^6 8.4784*^6 8.1471*^6
5.2381*^6 7.9052*^6
5.6222*^6

9.4608*^6 11.1832*^6 16.5121*^6 4.9891*^6 3.8514*^6 1.6696*^6 810692 334439 763229 0
5.3105*^6 10.7390*^6 7.9829*^6 3.4418*^6 2.5046*^6 4.2342*^6 2.9585*^6 461821 310645
4.5598*^6 11.6015*^6 7.7587*^6 5.5683*^6 1.7766*^6 3.3758*^6 1.9560*^6 332471
6.9037*^6 7.8402*^6 5.6170*^6 3.6026*^6 1.9997*^6 2.6543*^6 975817
3.6933*^6 7.7758*^6 7.0865*^6 5.7651*^6 3.9397*^6 606276
6.0641*^6 7.9475*^6 6.0275*^6 3.9985*^6 2.8606*^6
6.5412*^6 7.1789*^6 7.2484*^6 5.3578*^6
5.3251*^6 10.6930*^6 7.2173*^6
5.2515*^6 10.0257*^6
4.5726*^6

```
6.2652*^6 10.1288*^6  9.3906*^6  8.8062*^6 4.4417*^6   667233    724064   592094  366553  0
5.5643*^6  9.8488*^6 12.1532*^6 4.2317*^6 2.2264*^6 4.2392*^6 1.3689*^6 669875  305447
6.2595*^6 12.2046*^6  6.2927*^6  5.5831*^6 1.3445*^6   553408   1.0950*^6 479577
5.5420*^6  6.7224*^6  6.4654*^6  3.8741*^6 2.8154*^6 4.8073*^6 1.4972*^6
5.0441*^6  7.6390*^6  6.7167*^6  3.3955*^6 4.5529*^6 1.5408*^6
5.1999*^6  8.2514*^6  4.8507*^6  6.7869*^6 3.1742*^6
4.9219*^6  7.8540*^6  7.0257*^6  4.6842*^6
5.0892*^6 11.4150*^6  5.7817*^6
5.6259*^6  8.7293*^6
5.1640*^6


7.7935*^6 10.7650*^6 11.0188*^6 7.3320*^6 4.0450*^6 1.4763*^6 1.7922*^6 519534  345668  0
4.0832*^6  9.8264*^6 10.5979*^6 4.8170*^6 1.1518*^6 5.2284*^6   444730   375774  395630
6.1429*^6  9.3194*^6 11.1039*^6 4.3898*^6 2.3922*^6 1.3275*^6 1.7393*^6 724641
5.9395*^6  6.9865*^6  8.9202*^6  3.4109*^6 2.4671*^6 2.5154*^6 1.7959*^6
4.6156*^6  7.6134*^6  2.8383*^6  4.6732*^6 5.6115*^6   256786
5.5958*^6  8.3149*^6  4.9289*^6  5.8500*^6 1.1936*^6
4.6941*^6  7.6974*^6  7.7809*^6  4.3727*^6
4.8172*^6  9.2572*^6  7.3617*^6
6.3515*^6  9.4670*^6
4.3554*^6


6.1547*^6 12.8920*^6 15.4287*^6 10.1078*^6 4.0708*^6   880567   313716   706033   315076  0
4.0359*^6  9.1818*^6 12.5192*^6  5.1169*^6   557479   3.7618*^6 446149 1.6650*^6 348246
5.7528*^6  9.2164*^6  8.3265*^6  2.9830*^6  3.8327*^6 1.7987*^6 721737   324279
4.7092*^6  8.3045*^6  7.3404*^6  3.2990*^6  9.2939*^6 3.0878*^6 531529
5.1289*^6  5.3320*^6  3.9618*^6  1.1368*^6  6.7283*^6 4.8187*^6
5.0267*^6  7.1534*^6  5.6136*^6  3.0235*^6  2.6230*^6
5.9654*^6  8.3771*^6  6.7381*^6  5.9539*^6
4.7826*^6  5.6837*^6  7.7179*^6
5.4634*^6  7.5736*^6
4.7320*^6


6.6801*^6 11.1202*^6 10.5611*^6 7.6325*^6 3.8118*^6 1.4054*^6   324616   816188  308258  0
3.9932*^6  9.4867*^6 11.0689*^6 7.8441*^6 2.6446*^6 2.6983*^6 1.0217*^6 303079  314588
6.2948*^6 10.2981*^6 11.7709*^6 2.0020*^6 1.8008*^6 1.4388*^6   302825   505255
6.6034*^6  7.1420*^6  7.2142*^6  3.9685*^6 1.5207*^6 2.4519*^6 1.3394*^6
4.2450*^6  8.3622*^6  4.7548*^6  3.7860*^6 2.8149*^6 2.7683*^6
5.6203*^6  7.6386*^6  8.2439*^6  7.4350*^6 1.8720*^6
4.2885*^6  9.1757*^6  4.3229*^6  6.5603*^6
4.0822*^6  6.8575*^6  6.0199*^6
6.1056*^6 11.7731*^6
3.8768*^6


6.8997*^6 12.1181*^6 12.2209*^6 7.9462*^6 2.9206*^6 2.7002*^6   328487   364437  931225  0
3.7208*^6  9.5944*^6  8.0419*^6  7.5810*^6 1.7227*^6 5.7487*^6 1.6680*^6 333386  491252
5.9051*^6 10.4741*^6 12.0107*^6 4.5170*^6 1.1699*^6 1.7621*^6   820607   402679
5.8536*^6  7.8609*^6  6.1318*^6  1.7292*^6 2.4270*^6 1.8048*^6 1.4396*^6
4.2206*^6  9.5018*^6  4.5146*^6  4.4412*^6 6.6582*^6 2.0170*^6
5.4268*^6  7.0663*^6  5.8857*^6  5.5089*^6 1.6587*^6
4.0279*^6  8.9823*^6  6.3022*^6  6.2163*^6
5.0235*^6 16.2947*^6  6.2846*^6
4.9058*^6  7.0912*^6
4.9240*^6
```

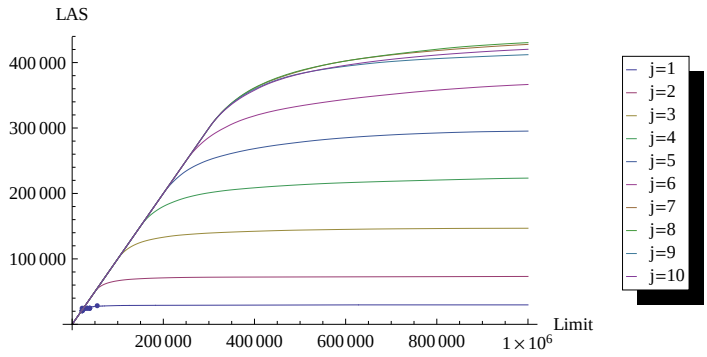
```
6.8965*^6 13.2339*^6 12.5568*^6 13.5930*^6 4.7779*^6 1.0002*^6 1.4372*^6 905926 1.1995*^6 0
4.0248*^6 10.5930*^6 8.8856*^6 5.0738*^6 2.5574*^6 2.0353*^6 642239 411036 310958
6.8032*^6 16.0075*^6 6.6417*^6 4.2668*^6 1.0457*^6 3.4208*^6 1.1218*^6 394321
6.8849*^6 7.2224*^6 5.4714*^6 4.0917*^6 252560 1.6863*^6 1.5327*^6
3.6526*^6 6.6504*^6 4.9115*^6 5.5113*^6 3.5094*^6 1.8526*^6
6.1610*^6 10.0142*^6 5.3851*^6 5.9940*^6 2.8616*^6
5.0903*^6 8.2828*^6 3.5513*^6 4.8768*^6
4.2429*^6 11.3124*^6 8.06264*^6
5.7175*^6 8.6900*^6
6.1056*^6


5.9927*^6 11.4042*^6 15.3298*^6 9.5871*^6 2.4205*^6 1.6122*^6 2.0804*^6 371121 715486 0
5.1132*^6 11.7868*^6 7.5377*^6 6.7651*^6 3.9428*^6 537402 758786 317305 407987
6.3664*^6 9.7800*^6 9.5764*^6 3.9180*^6 2.4755*^6 1.2136*^6 2.1095*^6 337682
6.3317*^6 9.7211*^6 7.9692*^6 1.9207*^6 1.6035*^6 2.0814*^6 434298
5.0492*^6 6.9189*^6 5.8487*^6 3.7264*^6 2.1577*^6 1.6987*^6
5.9913*^6 10.4522*^6 5.2674*^6 6.6768*^6 2.4644*^6
5.4170*^6 8.7150*^6 5.0797*^6 4.7041*^6
4.5885*^6 7.2183*^6 7.8792*^6
4.3119*^6 9.1388*^6
4.7967*^6


7.0031*^6 11.3051*^6 11.2564*^6 12.3410*^6 3.2274*^6 847265 376023 763893 724908 0
5.2142*^6 9.9538*^6 9.9887*^6 8.0601*^6 3.3965*^6 4.5525*^6 695415 1.1028*^6 544046
4.9345*^6 9.4526*^6 9.9180*^6 3.4704*^6 1.3223*^6 589486 1.3017*^6 1.1388*^6
5.5499*^6 8.0901*^6 6.2591*^6 2.1480*^6 1.3823*^6 2.9427*^6 666899
3.8561*^6 6.5937*^6 6.6366*^6 2.0860*^6 5.2740*^6 1.2086*^6
5.6601*^6 6.9004*^6 3.4815*^6 4.2754*^6 2.3478*^6
6.6013*^6 7.2457*^6 12.5206*^6 5.3792*^6
5.6424*^6 10.0119*^6 3.9897*^6
4.6362*^6 11.4353*^6
5.3312*^6


6.4364*^6 11.2016*^6 11.2496*^6 8.4267*^6 3.9100*^6 746450 1.4977*^6 434930 847624 0
5.4467*^6 9.2108*^6 11.2335*^6 4.2240*^6 1.9910*^6 1.8161*^6 442037 312966 300277
5.4903*^6 9.1302*^6 5.9639*^6 4.6818*^6 2.5744*^6 1.1616*^6 768788 322391
7.1286*^6 6.7645*^6 6.9916*^6 1.8314*^6 1.9109*^6 1.1458*^6 308677
4.6899*^6 10.5975*^6 7.6265*^6 2.9391*^6 2.3043*^6 2.0982*^6
6.2577*^6 8.7733*^6 9.0619*^6 4.8250*^6 2.9457*^6
4.5679*^6 6.6772*^6 8.0545*^6 8.4270*^6
6.0721*^6 7.4815*^6 9.2107*^6
4.9287*^6 8.3482*^6
5.0366*^6


7.6587*^6 11.4552*^6 14.0438*^6 7.9577*^6 4.2237*^6 797781 668352 547183 716439 0
5.2071*^6 8.8518*^6 9.9998*^6 6.4980*^6 4.0154*^6 3.0484*^6 3.1483*^6 832051 472192
6.1072*^6 10.3335*^6 6.2743*^6 2.8169*^6 1.0621*^6 2.1471*^6 2.2433*^6 463323
6.9107*^6 7.6230*^6 8.1744*^6 2.0962*^6 3.6330*^6 5.6393*^6 477294
4.2929*^6 5.7132*^6 4.0370*^6 1.9981*^6 4.6708*^6 1.5928*^6
5.1033*^6 7.7271*^6 5.0392*^6 6.5783*^6 1.7404*^6
6.7441*^6 9.3819*^6 7.8298*^6 8.4130*^6
5.6520*^6 12.2814*^6 6.2099*^6
5.4687*^6 10.5495*^6
4.3251*^6
```

Generator vyvojovych trojuholnikov

```
(* maxT udava pocet trojuholnikov, ktore chceme vygenerovat*)
maxT = 15;

(* THETA a ALPHA su dane parametre Paretovho
 rozdelenia a funkcia Pareto generuje vysky skod *)
THETA = {10 000, 25 000, 50 000, 75 000, 100 000,
   125 000, 150 000, 150 000, 150 000, 150 000, 150 000};
ALPHA = ConstantArray[2, 10];
Pareto = Map[ParetoDistribution[THETA[[#]], ALPHA[[#]]] &, Range[10]];

(* lambdaIJ a c su parametre negativneho binomickeho rozdelenia *)
lambdaIJ =
  {{254.666, 158.933, 88.405, 37.243, 12.451, 3.657, 0.791, 0.512, 0.450, 0.000},
   {169.600, 124.103, 65.066, 27.155, 8.742, 8.146, 1.694, 0.519, 0.000},
   {206.968, 134.641, 55.047, 17.712, 4.984, 2.764, 1.706, 0.118},
   {185.884, 100.266, 46.769, 13.700, 5.688, 6.132, 1.403},
   {148.800, 93.482, 36.949, 13.601, 12.832, 3.649},
   {175.288, 114.261, 37.795, 26.306, 6.723}, {175.502, 104.632, 41.365, 25.021},
   {169.315, 119.224, 46.705}, {178.666, 126.549}, {171.520}};
c = 0.01;

(* KMR modeluje uhrn skod pre jedno i,j *)
KMR[lambda_, j_] := Module[
   {dist, N, X, Zs},
   dist = NegativeBinomialDistribution[1 / c, 1 / (1 + c * lambda)];
   N = Round[RandomVariate[dist]] + 1;
   Zs = Table[RandomVariate[Pareto[[j]]] + THETA[[j]], {i, N}];
   X = Total[Zs];
   X
  ];

(* GenerujT vygeneruje jeden vyvojovy trojuholnik *)
GenerujT := Module[
   {T},
   T = Table[
     KMR[lambdaIJ[[i, j]], j],
     {i, 10}, {j, 1, 11 - i}
    ]; T[[1, 10]] = 0;
   T
  ];

Trojuholniky = Table[GenerujT, {i, maxT}]
```

Funkcia LAS

```
ThetaAlpha = {{10 000, 2}, {25 000, 2}, {50 000, 2}, {75 000, 2}, {100 000, 2},
   {125 000, 2}, {150 000, 2}, {150 000, 2}, {150 000, 2}, {150 000, 2}};
maxV = 640;

(* Pareto vygeneruje Paretovo rozdelenie pre dane parametre *)
Pareto = Map[ParetoDistribution[ThetaAlpha[[#]][[1]], ThetaAlpha[[#]][[2]]] &,
   Range[10]];

(* VyskySkod vyberie nÆhodnu vzorku z Paretovho rozdelenia,
APareto prevedie na americku definiciu Paretovho rozdelenia,
SeedRandom zaruci,
ze pri kazdom spusteni programu vygenerujeme rovnake vysky skod *)
SeedRandom["Kosova"];
VyskySkod = Map[RandomVariate[Pareto[[#]], maxV] &, Range[10]];
APareto[l_, d_] := Map[# + d &, l];
VyskySkod = Map[APareto[VyskySkod[[#]], ThetaAlpha[[#]][[1]]] &, Range[10]];

LAS[j_, x_] := Module[{vzorka},
   vzorka = Map[Min[x, #] &, VyskySkod[[j]]]; Total[vzorka] / Length[vzorka]];
Needs["PlotLegends`"]
Plot[Evaluate[Table[LAS[j, x], {j, 10}]], {x, 0, 1 000 000},
 AxesLabel → {"Limit", "LAS"}, Background → White,
 PlotLegend → {"j=1", "j=2", "j=3", "j=4", "j=5", "j=6",
   "j=7", "j=8", "j=9", "j=10"}, LegendPosition → {1.1, -0.4}]
```



CNB model

Funkcia LAS

```
ThetaAlpha = {{10 000, 2}, {25 000, 2}, {50 000, 2}, {75 000, 2}, {100 000, 2},
   {125 000, 2}, {150 000, 2}, {150 000, 2}, {150 000, 2}, {150 000, 2}};
maxV = 640;
Pareto = Map[ParetoDistribution[ThetaAlpha[[#]][[1]], ThetaAlpha[[#]][[2]]] &,
   Range[10]];
SeedRandom["Kosova"];
VyskySkod = Map[RandomVariate[Pareto[[#]], maxV] &, Range[10]];
APareto[l_, d_] := Map[# + d &, l];
VyskySkod = Map[APareto[VyskySkod[[#]], ThetaAlpha[[#]][[1]]] &, Range[10]];
LAS[j_, x_] := Module[{vzorka},
   vzorka = Map[Min[x, #] &, VyskySkod[[j]]]; Total[vzorka] / Length[vzorka]];
```

Vyvojove trojuholniky

```
Trojuholniky = {(* Vygenerovane trojuholniky *)};
```

Diskretizacia

```
Poistne = {50 000 000, 50 000 000, 50 000 000, 50 000 000,
   50 000 000, 50 000 000, 50 000 000, 50 000 000, 50 000 000, 50 000 000};
PolicyLimit = 1 000 000;
rho = 2^14;
M = {5, 10, 15, 20, 25, 40, 50, 100, 125, 200, 250, 500, 1000};
h' = Total[Poistne] / rho;
h = Min[Select[M, # ≥ h' / 1000 &]] * 1000;
k = PolicyLimit / h;

(* inicializacia a vypocet diskretizovanych pravdepodobnosti p *)
p = Table[0, {i, rho}, {j, 10}];
For[i = 1, i ≤ rho, i++,
  For[j = 1, j ≤ 10, j++,
   If[i == 1, p[[i, j]] = 1 - LAS[j, h] / h; Continue[]];
   If[i ≤ k, p[[i, j]] =
     (2 * LAS[j, h * (i - 1)] - LAS[j, (i - 2) * h] - LAS[j, i * h]) / h; Continue[]];
   If[i == k + 1, p[[i, j]] = 1 - Sum[p[[x, j]], {x, 1, k}]; Continue[]];
   p[[i, j]] = 0;]
 ];
pT = Transpose[p];
```

Odhad parametrov

```
(* EZ - stredna hodnota vysky skod, americke Pareto *)
EZ = Table[2^(-ThetaAlpha[[j, 2]]) * (1 + ThetaAlpha[[j, 2]]) *
    ThetaAlpha[[j, 1]] / (ThetaAlpha[[j, 2]] - 1), {j, 1, 10}];

(* FFT diskretizovanych pravdepodobnosti *)
PsiP = Map[Fourier[#] &, pT];

(* Cape Cod metoda *)
CC[t_] := Table[Poistne[[i]] * t[[11]] * t[[j]], {i, 1, 10}, {j, 1, 11 - i}];

(* maximalizacne body, podmienky maximalizacie,
pociatocny bod optimalizacie *)
xs = {x1, x2, x3, x4, x5, x6, x7, x8, x9, x10, x11};
podmienky = {x1 > 0.1, x2 > 0.2, x3 > 0.1, x4 > 0, x5 > 0, x6 > 0, x7 > 0,
   x8 > 0, x9 > 0, x10 > 0, x11 > 0.4, x11 < 0.9, x1 < x2, x2 > x3, x3 > x4,
   x4 > x5, x5 > x6, x6 > x7, x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 == 1,
   x8 < 0.5 * x7, x9 < 0.5 * x8, x10 < 0.5 * x9};
bod = {0.16801578268304668', 0.27354897383129184', 0.23909072139186907',
   0.1575745222926144', 0.07841733760303848', 0.05569662313336992',
   0.018956200943932168', 0.006102673477421544', 0.001964667948988886',
   0.0006324966944270059', 0.6147753731108555'};

(* ODPVierohodnost a maxODP vypocita pociatocny vektor Nelder-
 Meada pre CNB likelihood *)
ODPVierohodnost[ X_, s_] := Module[
   {cc},
   cc = CC[s];
   Sum[X[[i, j]] * Log[Abs[cc[[i, j]]]] - Abs[cc[[i, j]]],
    {i, 1, 10}, {j, 1, 11 - i}]
  ];

(* CNB pomocou FFT vypocita podmienenu pravdepodobnost uhrnu skod *)
CNB[x_, lambda_, psip_] := Module[
   {psiQ, Q, i},
   psiQ = (1 - 0.01 * lambda * (Sqrt[rho] * psip - 1))^(-100);
   Q = InverseFourier[psiQ] / Sqrt[rho];
   i = IntegerPart[x / h] + 1;
   Q[[i]]
  ];

(* CNBVierohodnost a maxCNB vypocita najlepsie odhady neznamych parametrov*)
CNBVierohodnost[X_, s_?(VectorQ[#, NumericQ] &)] := Module[
   {cc, lambda},
   cc = CC[s];
   lambda = Table[cc[[i, j]] / EZ[[j]], {i, 10}, {j, 1, 11 - i}];
   Product[
    CNB[X[[i, j]], lambda[[i, j]], PsiP[[j]]],
    {i, 10}, {j, 1, 11 - i}]
  ];
```

```
(*Funkcia Ries odhadne parametre postupne pre vsetky trojuholniky*)
Ries[T_] := Module[
   {maxODP, najODP, maxCNB},
   SetOptions[NMaximize, MaxIterations → 100];
   maxODP = NMaximize[
     {ODPVierohodnost[T, xs], podmienky},
     xs,
     Method → {"NelderMead", InitialPoints → {bod}}];
   najODP = Flatten[xs /. Rest[maxODP]];
   SetOptions[NMaximize, MaxIterations → 30];
   maxCNB = NMaximize[
     {CNBVierohodnost[T, xs], podmienky},
     xs,
     Method → {"NelderMead", InitialPoints → {najODP}}];
   najCNB = Flatten[xs /. Rest[maxCNB]];
   najCNB
  ];

(* odhadnute parametre *)
parametre = Map[Ries[#] &, Trojuholniky]
```

Mnoziny parametrov

```
Parametre = parametre;

Trojuholnik = {(* Trojuholnik poistovne A *)};

(* Funkcia vytvara mnozinu sad parametrov Omega *)
Rozsir[par_] := Module[
   {Ls, maxBeta, najBeta, EL, KROK},
   Ls = Map[#[[11]] &, par];
   maxBeta = NMaximize[{Likelihood[BetaDistribution[a, b], Ls],
      a > 0, b > 0, a < 100, b < 100}, {a, b}];
   najBeta = Flatten[{a, b} /. maxBeta[[2]]];
   EL = Mean[BetaDistribution[najBeta[[1]], najBeta[[2]]]];
   KROK = 0.03;
   Ls = Table[EL + i * KROK, {i, -4, 4}];
   Flatten[
    Table[Flatten[{Take[par[[i]], 10], Ls[[j]]}], {i, Length[par]}, {j, 9}], 1]
  ];
```

Bayes

```
(* Najdeme sadu parametrov, pre ktore je vierohodnostna
 funkcia CNB rozdelenia pre Trojuholnik maximalna *)
CC[t_] := Module[{}, Table[Poistne[[i]] * t[[11]] * t[[j]], {i, 10}, {j, 10}]];
CNB[x_, lambda_, psip_] := Module[
   {psiQ, Q, i, c},
   psiQ = (1 - 0.01 * lambda * (Sqrt[rho] * psip - 1))^(-100);
   Q = InverseFourier[psiQ] / Sqrt[rho];
   i = IntegerPart[x / h] + 1;
   Q[[i]]
  ];
CNBVierohodnost[X_, s_?(VectorQ[#, NumericQ] &)] := Module[
   {cc, lambda},
   cc = CC[s];
   lambda = Table[cc[[i, j]] / EZ[[j]], {i, 10}, {j, 1, 11 - i}];
   Product[CNB[X[[i, j]], lambda[[i, j]], PsiP[[j]]], {i, 10}, {j, 1, 11 - i}]
  ];

(* spocita Bayesovu vetu, likelihood pre kazdu sadu parametrov *)
Bayes[X_, Omega_] := Map[{CNBVierohodnost[X, #], #} &, Omega]

(* vyberie s pomedzi parametrov tie, pre ktore ma Bayes najvyssiu hodnotu *)
maxBy[list_, n_] := With[{s = list[[All, n]]}, Pick[list, s, Max[s]]];

Omega = Rozsir[Parametre];
likelihood = Bayes[Trojuholnik, Omega];
najParametre = maxBy[likelihood, 1][[1]][[2]]

(* Nakoniec spocitame rezervu *)
cc = CC[najParametre];
Rezerva = Table[
   If[i + j ≤ 11, Trojuholnik[[i, j]], cc[[i, j]]],
   {i, 10}, {j, 10}];
Grid[%]

RezervaCelkom = Sum[Rezerva[[i, j]], {i, 2, 10}, {j, 12 - i, 10}]
```

Chain-Ladder

Nekumulativny trojuholnik -> kumulativny trojuholnik

```
Trojuholnik = {{7.168*^6, 1.119*^7, 1.2432*^7,
    7.856*^6, 3.502*^6, 1.286*^6, 334 000, 216 000, 190 000, 0},
   {4.77*^6, 8.726*^6, 9.15*^6, 5.728*^6, 2.459*^6,
    2.864*^6, 715 000, 219 000, 0},
   {5.821*^6, 9.467*^6, 7.741*^6, 3.736*^6, 1.402*^6, 972 000, 720 000, 50 000},
   {5.228*^6, 7.05*^6, 6.577*^6, 2.89*^6, 1.6*^6, 2.156*^6, 592 000},
   {4.185*^6, 6.573*^6, 5.196*^6, 2.869*^6, 3.609*^6, 1.283*^6},
   {4.93*^6, 8.034*^6, 5.315*^6, 5.549*^6, 1.891*^6},
   {4.936*^6, 7.357*^6, 5.817*^6, 5.278*^6},
   {4.762*^6, 8.383*^6, 6.568*^6},
   {5.025*^6, 8.898*^6},
   {4.824*^6}};
KT = Table[0, {i, 10}, {j, 11 - i}];
KumT[trojuholnik_] :=
  Module[{pom, j, i}, For[i = 1, i <= 10, i++, pom = 0; For[j = 1, j ≤ 11 - i,
     j++, pom = trojuholnik[[i, j]] + pom; KT[[i, j]] = pom]]; KT];
KumTrojuholnik = KumT[Trojuholnik];
```

Vyvojove koeficienty

```
f = Table[0, {j, 9}];
VyvojF[trojuholnik_] := Module[{i, j}, For[j = 1, j ≤ 9, j++,
                                f[[j]] = (Sum_{i=1}^{10-j} trojuholnik[[i, j + 1]]) / (Sum_{i=1}^{10-j} trojuholnik[[i, j]])]; f];
VyvojoveKoeficienty = VyvojF[KumTrojuholnik];
```

Kumulativne uhrny skod

```
X = Table[0, {i, 9}];
pomF := Module[{pom = 1, j},
  For[j = 1, j ≤ 9, j++, pom = VyvojoveKoeficienty[[10 - j]] * pom;
   VyvojoveKoeficienty[[10 - j]] = pom]; VyvojoveKoeficienty]
Uhrny[trojuholnik_, f_] := Module[{}, For[i = 2, i ≤ 10, i++,
    X[[i - 1]] = trojuholnik[[i, 11 - i]] * f[[11 - i]]]; X];
KumUhrny = Uhrny[KumTrojuholnik, pomF];
```

Rezerva

```
pom = Sum[KumTrojuholnik[[i, 11 - i]], {i, 2, 10}];
Rezerva = Total[KumUhrny] - pom
```